TOUGH+HYDRATE v1.2
USER'S MANUAL:
A CODE FOR THE SIMULATION OF SYSTEM BEHAVIOR IN HYDRATE-BEARING GEOLOGIC MEDIA

George J. Moridis, Michael B. Kowalsky, and Karsten Pruess

Earth Sciences Division,
Lawrence Berkeley National Laboratory,
Berkeley, CA 94720

August 2012

This work was supported by the Assistant Secretary for Fossil Energy, Office of Natural Gas and Petroleum Technology, through the National Energy Technology Laboratory, under the U.S. Department of Energy, Contracts No. DE-AC03-76SF00098 and DE-AC02-05CH11231.

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor The Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or The Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or The Regents of the University of California.

Ernest Orlando Lawrence Berkeley National Laboratory is an equal opportunity employer.

TOUGH+HYDRATE v1.2 User's Manual: A Code for the Simulation of System Behavior in Hydrate-Bearing Geologic Media

George J. Moridis, Michael B. Kowalsky, and Karsten Pruess
Earth Sciences Division, Lawrence Berkeley National Laboratory
University of California, Berkeley, California

Abstract

TOUGH+HYDRATE v1.2 is a code for the simulation of the behavior of hydrate-bearing geologic systems, and represents the second update of the code since its first release [*Moridis et al*., 2008]. By solving the coupled equations of mass and heat balance, TOUGH+HYDRATE can model the non-isothermal gas release, phase behavior and flow of fluids and heat under conditions typical of common natural CH_4-hydrate deposits (i.e., in the permafrost and in deep ocean sediments) in complex geological media at any scale (from laboratory to reservoir) at which Darcy's law is valid.

TOUGH+HYDRATE v1.2 includes both an equilibrium and a kinetic model of hydrate formation and dissociation. The model accounts for heat and up to four mass components, i.e., water, CH_4, hydrate, and water-soluble inhibitors such as salts or alcohols. These are partitioned among four possible phases (gas phase, liquid phase, ice phase and hydrate phase). Hydrate dissociation or formation, phase changes and the corresponding thermal effects are fully described, as are the effects of inhibitors. The model can describe all possible hydrate dissociation mechanisms, i.e., depressurization, thermal stimulation, salting-out effects and inhibitor-induced effects.

TOUGH+HYDRATE is a member of TOUGH+, the successor to the TOUGH2 [*Pruess et al.*, 1991] family of codes for multi-component, multiphase fluid and heat flow developed at the Lawrence Berkeley National Laboratory. It is written in standard FORTRAN 95/2003, and can be run on any computational platform (workstation, PC, Macintosh) for which such compilers are available.

TABLE OF CONTENTS

Abstract iii

List of Figures ix

List of Tables xiii

1.0. INTRODUCTION 1

1.1. Background 1

1.2. TOUGH+HYDRATE v1.0 Code 3

2.0 CONCEPTS, UNDERLYING PHYSICS, AND GOVERNING EQUATIONS .. 9

2.1. Modeled Processes and Underlying Assumptions 9

2.2. Components and Phases 12

2.3. The Mass and Energy Balance Equation 14

2.4. Mass Accumulation Terms 14

2.5. Heat Accumulation Terms 17

2.6. Flux Terms 19

2.7. Source and Sink Terms 22

2.8. Thermophysical Properties 24
2.8.1. Water *24*
2.8.2. CH_4-Hydrate *24*
2.8.3. CH_4 Gas *25*

2.9. Hydrate Phase Relationships 26

2.10. Inhibitor Effects on Hydrate Equilibrium 28

2.11. Porosity and intrinsic permeability change 34

2.12. Wettability Phenomena 37
2.12.1. Capillary pressure *37*
2.12.2. Relative Permeability *41*
2.12.3. Pore compressibility of unconsolidated media in the presence of cementing solid phases *49*

2.13. Multiphase Diffusion 53
2.13.1. General Considerations *53*
2.13.2. Diffusion Formulation in TOUGH+ *54*

2.14. Description of Flow in Fractured Media 55

3.0. DESIGN AND IMPLEMENTATION OF TOUGH+HYDRATE 59

3.1. Primary Variables **59**

3.2. Space and Time Discretization **63**

3.3. The Newton-Raphson Iteration **65**

3.4. Implications of the Space Discretization Approach **66**

3.5. Space Discretization of Diffusive Fluxes **67**

3.6. Compiling the TOUGH+HYDRATE Code **69**

4.0. INPUT DATA REQUIREMENTS 75

4.1. Input Procedure **75**
4.1.1. Data Block/Keyword `TITLE` *76*
4.1.2. Keyword/Record `START` or `RANDOMN` *76*
4.1.3. Keyword/Record `ENDCY` *77*
4.1.4. Keyword/Record `ENDFI` *77*
4.1.5. General Comments on the Structure of TOUGH+ Standard Input Files *77*

5.0. MEMORY SPECIFICATION AND ALLOCATION 81

5.1. Data Block `MEMORY` **81**

6.0. PHYSICAL PROPERTIES OF SYSTEM 91

6.1. Data Block `HYDRATE` **91**

6.2. Data Block `ROCKS` or `MEDIA` **94**

6.3. Data Block `RPCAP` **100**
6.3.1. Relative Permeability Functions *101*
6.3.2. Capillary Pressure Functions *104*

6.4. Data Block `DIFFUSION` **107**
6.4.1. User Options for Multiphase Diffusion *110*

6.5. Block-by-Block Permeability Modification **112**

7.0. GEOMETRICAL REPRESENTATION, DOMAIN DISCRETIZATION, AND GRID GENERATION 115

7.1. TOUGH+ Convention for Geometrical Data **115**

7.2. Data Block `ELEME` **117**

7.3. Data Block `CONNE` **120**

7.4. The MeshMaker.f95 Facility **122**
7.4.1. Inputs Related to Problem Definition and Dimensioning *122*
7.4.2. Inputs Related to Domain Heterogeneity *124*
7.4.3. Inputs Related to Description of Boundaries *127*
7.4.4. Inputs for Grid Construction *130*

8.0. INITIAL CONDITIONS AND BOUNDARY CONDITIONS **143**

8.1. Data Block INCON **143**

8.2. Data Block INDOM **146**

8.3. Data Block EXT-INCON **147**
8.3.1. Data Block GEOMETRY *148*
8.3.2. Data Block SEQUENCE *149*
8.3.3. Data Block LIST *150*
8.3.4. Remaining Data Blocks in EXT-INCON *151*
8.3.5. Data Block COLUMN *151*

8.4. Implementing Initial Conditions **153**

8.5. Implementing Boundary Conditions **155**

9.0. SOURCES AND SINKS **157**

9.1. Data Block GENER **157**

9.2. Discussion on sinks and sources **162**
9.2.1. Deliverability Model *163*
9.2.2. Coupled Wellbore Flow *165*

10. COMPUTATIONAL PARAMETERS **169**

10.1. Data Block PARAM **169**

10.2. Modification of Computational Parameters During the Course of a TOUGH+ Simulation **183**

10.3. Data Block SOLVR **187**

10.4. Discussion on Linear Equation Solvers **189**

11. OUTPUT SPECIFICATIONS **191**

11.1 Output of Primary and Secondary Variables **192**

11.2. Data Block TIMES **193**

11.3. Data Block Elem_Time_Series or FOFT **194**

11.4. Data Block Conx_Time_Series or COFT **195**

11.5. Data Block SS_Time_Series or GOFT 197

11.6. Data Block SUBDOMAINS 198

11.7. Data Block NOVER 202

11.8. Warning Output and Error Messages 202

12.0. EXAMPLE PROBLEMS 204

12.1. Example Files and Naming Conventions 204

12.2. Problem Test_1T: Thermal Stimulation, Equilibrium Dissociation, No Inhibitor 205

12.3. Problem Test_1Tk: Thermal Stimulation, Kinetic Dissociation, No Inhibitor 211

12.4. Problems Test_1P: Depressurization, Equilibrium Dissociation, No Inhibitor 220

12.5. Problem Test_1Pk: Depressurization, Kinetic Dissociation, No Inhibitor 222

12.6. Problem Test_1P_Ice: Thermal Stimulation, Kinetic Dissociation, No Inhibitor, Ice Evolution 223

12.7. Problem Test_1TbS and Test_1TbSk: Hydrate Formation, Equilibrium and Kinetic Hydration reaction, Inhibitor 231

12.8. Problem Test_2Qp: Equilibrium Hydrate Dissociation, Depressurization, Radial Grid - Single Well 235

12.9. Problem Test_3Qpk: Kinetic Hydrate Dissociation, Depressurization, Radial Grid - Single Well with Wellbore Heating 239

12.10. Problem Test_2D: Equilibrium Hydrate Dissociation, 2-D Areal System 243

Acknowledgements 252

References 254

Appendix A 264

A1. Input File: Test_1T, Output File: Hydrate_Info_1T 264

A2. Input File: Test_1T, Output File: Conx_Time_Series_1T 266

A3. Input File: Test_1P_Ice, Output File: Hydrate_Info_1P_Ice 267

List of Figures

Figure 2.1. Relationship of the equilibrium hydration pressure *Pe* and temperature *Te* of the CH_4-hydrate [*Moridis,* 2003]. 29

Figure 2.2. Modified *Pe* -*Te* relationship in the vicinity of the quadruple point. 30

Figure 2.3. Temperature dependence of the CH_4-hydrate heat of dissociation ΔH^0 along the (a) I-H-V and (b) L_w -H-V phase lines [*Moridis*, 2003]. 31

Figure 2.4. Modified ΔH^0 vs. T*e* relationship used in TOUGH+HYDRATE. 32

Figure 2.5. Pressure-temperature equilibrium relationship in the phase diagram of the water--CH4--hydrate system in T+H. 33

Figure 2.6. Schematic of pore channels, showing convergent-divergent geometry with a succession of pore throats and pore bodies. 35

Figure 2.7. Tubes-in-series model of pore channels. 37

Figure 2.8. Schematic of probability density function p(r) for pore size distribution. 38

Figure 2.9. Liquid and gas relative permeabilities based on the van Genuchten model [*Finsterle*, 1999]. 44

Figure 2.10. Compressibility of an unconsolidated porous medium impregnated with cementing solid phases (ice and/or hydrates). In this example, S_{Smin} = 0.15, S_{Smax} = 0.4, α_{PU} = 10^{-8} Pa^{-1}, α_{PL} = 10^{-9} Pa^{-1} and δ = 0.015. 51

Figure 2.11. Effect of the varying compressibility described in Figure 2.10 on the porosity of an unconsolidated porous medium undergoing depressurization for various levels of saturation S_S of cementing solid phases. 52

Figure 2.12. Idealized double porosity model of a fractured porous medium. 57

Figure 2.13. Subgridding in the method of "multiple interacting continua" (MINC). 57

Figure 2.14. Flow connections in the “dual permeability” model. Global flow occurs between both fracture (F) and matrix (M) grid blocks. In addition there is F-M interporosity flow. 59

Figure 3.1. Space discretization and geometry data in the integral finite difference method. 64

Figure 6.1. The **`DIFFUSION`** data block, with examples of the `Diffusion_Key_Parameters` and `Component_Diffusivities_in_Phases` namelists. 146

Figure 7.1. An example of a `MeshMaker.f95` input file for the creation of a Cartesian 3D grid. Note that no heterogeneous regions or boundaries are defined in this grid. 139

Figure 7.2. An example of a MeshMaker.f95 input file for the creation of a single-layer (1D) cylindrical grid. Note that no heterogeneous regions or boundaries are defined in this grid. 139

Figure 7.3. An example of a MeshMaker.f95 input file for the creation of a large Cartesian 3D grid with heterogeneous regions and defined boundaries. 140

Figure 7.4. An example of a MeshMaker.f95 input file for the creation of a large cylindrical 2D grid with multiple layers, heterogeneous regions and defined boundaries. 141

Figure 8.1. An example of the NAMELIST-described termination data printed at the end of the SAVE file from a TOUGH+ simulation. These data can be read as part of the INCON data block, or of the INCON external file. The names of the variables defined in the NAMELIST are self-explanatory. For reference, this figure lists the conditions in the last two elements (FJ000 and FJ106) of the grid in the TOUGH+ simulation. 146

Figure 10.1. An example of a Parameter_Update_File for parameter updating in the course of a TOUGH+ simulation. Within the namelists (Real_Parameters_To_Update and Integer_Parameters_To_Update), parameters can be entered in any order, data are read using any kind of appropriate format, only the needed parameters are included, and comments can be added. 185

Figure 10.2. An example of a Parameter_Update_File indicating three completed parameter updates, in addition to another one (at the top of the file) that has not yet been executed. 186

Figure 11.1. An example of the SUBDOMAINS data block for tracking the evolution of volume-averaged properties and conditions in specified subdomains. This data block uses namelist-based formats for data inputs. 201

Figure 12.1. Input file for example problem Test_1T (in Section 12.2) involving thermal stimulation, equilibrium dissociation, and no inhibitor. 210

Figure 12.2. Input file for example problem Test_1Tk (in Section 12.3) involving thermal stimulation, kinetic dissociation, and no inhibitor. 215

Figure 12.3. Comparison of the volumetric rates of CH_4 release from hydrate dissociation in problems Test_1T and Test_1Tk. 216

Figure 12.4. Comparison of the cumulative volumes of CH_4 released from hydrate dissociation in problems Test_1T and Test_1Tk. 217

Figure 12.5. Comparison of the volumetric production rates of CH_4 (measured as the flux crossing the x = 0 boundary) in problems Test_1T and Test_1Tk. 218

Figure 12.6. Comparison of the total production volumes of CH_4 (measured as the total volumes crossing the x = 0 boundary) in problems Test_1T and Test_1Tk. 219

Figure 12.7. Input file for example problem Test_1Pk (in Section 12.4) involving depressurization, equilibrium dissociation, and no inhibitor. 222

Figure 12.8. Comparison of the volumetric rates of CH_4 release from depressurization-induced hydrate dissociation in problems `Test_1P, Test_1Pk` and `Test_1P_ice`. 227

Figure 12.9. Comparison of the cumulative volumes of CH_4 released from depressurization-induced hydrate dissociation in problems `Test_1P, Test_1Pk` and `Test_1P_ice`. 228

Figure 12.10. Comparison of the volumetric production rates of CH_4 (measured as the flux crossing the x = 0 boundary) in problems `Test_1P, Test_1Pk` and `Test_1P_ice`. 229

Figure 12.11. Comparison of the total production volumes of CH_4 (measured as the total volumes crossing the x = 0 boundary) in problems `Test_1P, Test_1Pk` and `Test_1P_ice`. 230

Figure 12.12. Comparison of the volumetric rates of CH_4 consumption during hydrate formation in problems `Test_1TbS, Test_1TbSk` and `Test_1TbS2`. 233

Figure 12.13. Comparison of the cumulative volumes of CH_4 consumed during hydrate formation in problems `Test_1TbS, Test_1TbSk` and `Test_1TbS2`. 234

Figure 12.14. Volumetric rate of CH_4 release during hydrate dissociation in problem `Test_2Qp`. 237

Figure 12.15. Cumulative volume of CH_4 released during hydrate dissociation in problem `Test_2Qp`. 238

Figure 12.16. Volumetric rates of CH_4 (a) release in the reservoir during hydrate dissociation and (b) production from the well in problem `Test_3Qpk`. 241

Figure 12.17. Cumulative volumes of CH_4 (a) released in the reservoir during hydrate dissociation and (b) produced from the well in problem `Test_3Qpk`. 242

Figure 12.18. Volumetric rates of CH_4 release from the hydrate in the reservoir during gas production in problem `Test_2D`. 246

Figure 12.19. Pressure distribution in the reservoir at t = 2 days in problem `Test_2D`. 247

Figure 12.20. Temperature distribution in the reservoir at t = 2 days in problem `Test_2D`. 248

Figure 12.21. Hydrate saturation distribution in the reservoir at t = 2 days in problem `Test_2D`. 249

Figure 12.22. Aqueous phase saturation distribution in the reservoir at t = 2 days in problem `Test_2D`. 250

Figure 12.23. Gas saturation distribution in the reservoir at t = 2 days in problem `Test_2D`. 251

List of Tables

Table 3.1. Primary Variables in Equilibrium Hydrate Simulations without Inhibitor*..... 61

Table 3.2. Primary Variables in Kinetic Hydrate Simulations without Inhibitor*..... 62

Table 4.1. TOUGH+ input data blocks. 80

1.0. Introduction

1.1. Background

Gas hydrates are solid crystalline compounds in which gas molecules are encaged inside the lattices of ice crystals. These gases are referred to as guests, whereas the ice crystals are called hosts. Of particular interest are hydrates in which the gas is a hydrocarbon. Under suitable conditions of low temperature and high pressure, a hydrocarbon gas M will react with water to form hydrates according to

$$M + N_H H_2O = M \cdot N_H H_2O \tag{1.1}$$

where N_H is the hydration number.

Vast amounts of hydrocarbons are trapped in hydrate deposits [*Sloan*, 1998]. Such deposits exist where the thermodynamic conditions allow hydrate formation, and are concentrated in two distinctly different types of geologic formations where the necessary low temperatures and high pressures exist: in the permafrost and in deep ocean

sediments. The lower depth limit of hydrate deposits is controlled by the geothermal gradient.

Current estimates of the worldwide quantity of hydrocarbon gas hydrates range between 10^{15} to 10^{18} m^3.Even the most conservative estimates of the total quantity of gas in hydrates may surpass by a factor of two the energy content of the total fuel fossil reserves recoverable by conventional methods. The magnitude of this resource could make hydrate reservoirs a substantial future energy resource. While current economic realities do not favor gas production from the hydrate accumulations, their potential clearly demands evaluation.

The majority of naturally occurring hydrocarbon gas hydrates contain CH_4 in overwhelming abundance. Simple CH_4-hydrates concentrate methane volumetrically by a factor of 164, and require less than 15% of the recovered energy for dissociation. Natural CH_4-hydrates crystallize mostly in the I structure, which contains 46 H_2O molecules per unit cell. They have a N_H ranging from 5.77 to 7.41, with N_H = 6 being the average hydration number and N_H = 5.75 corresponding to complete hydration [*Sloan*, 1998].

There are three main methods of hydrocarbon recovery from gas hydrates: (a) thermal stimulation [*McGuire*, 1981], in which gas release is effected by heating the hydrate above the dissociation temperature at a given pressure, (b) depressurization [*Holder et al.*, 1982], in which the gas release is achieved by lowering the pressure below that of the hydrate stability, and (c) 'inhibitor' injection [*Kamath and Godbole*, 1987] in which the hydrocarbon is produced after the injection of substances (e.g., brines, alcohols) that destabilize the hydrate. Combinations of these methods can also be used.

Numerical codes and studies on the simulation of gas production from dissociating hydrates are limited. *Drenth and Swinkels* [2000] developed a four-component, three-phase numerical model for the equilibrium dissociation of binary hydrates in marine environments. They provided an in-depth discussion of the challenges facing production from gas hydrates and identified knowledge gaps in numerical simulation of gas production from hydrate dissociation. *Kurihara et al.* [2003] developed a numerical simulator, and used it for the prediction of gas production from gas hydrates from both marine and permafrost hydrate deposits.

Moridis et al. [1998] developed EOSHYDR, a TOUGH2 [*Pruess et al.*, 1999] module for the simulation of dissociating simple methane hydrates under equilibrium conditions in both permafrost and marine accumulations. *Moridis et al.* [2003] enhanced EOSHYDR and developed EOSHYDR2 – also a TOUGH2 module – for the simulation of binary hydrates reacting under both equilibrium and kinetic conditions. TOUGH2/EOSHYDR2 was used for the simulation of gas production from hydrates under a variety of geologic and thermodynamic conditions, and involving various production strategies [*Moridis*, 2003; *Moridis et al.*, 2004; *Moridis and Collett*, 2004].

1.2. TOUGH+HYDRATE v1.2 Code

TOUGH+HYDRATE v1.2 is a code for the simulation of the behavior of hydrate-bearing geologic systems, and is the second update of the code first released in 2008 [*Moridis et al.*, 2008]. TOUGH+HYDRATE (hereafter often referred to as **T+H** for brevity) is the first member of TOUGH+, the successor to the TOUGH2 [*Pruess et al.*, 1991] family of codes for multi-component, multiphase fluid and heat flow developed at

the Lawrence Berkeley National Laboratory. It is written in standard FORTRAN 95/2003 to take advantage of all the object-oriented capabilities and the enhanced computational features of that language.

While the underlying principles, physics and thermodynamics are similar to those used in TOUGH2/EOSHYDR2 [*Moridis*, 2003], the code in **T+H** - and in the earlier, transitional version of the code named TOUGH-Fx/HYDRATE [*Moridis et al*., 2005], of which the **T+H** code is a natural extension - was completely rewritten and its architecture is drastically different. The new code employs dynamic memory allocation, thus minimizing storage requirements. It follows the tenets of Object-Oriented Programming (OOP), and involves entirely new data structures that describe the objects upon which the code is based. The basic objects are defined through derived data types, and their properties and processes are described in modules and sub-modules, i.e., entities that incorporate the object attributes and parameters in addition to the procedures (corresponding to the older concepts of "functions" and "subroutines" in FORTRAN 77) that describe its behavior and processes. The TOUGH+ code is based on a completely modular structure that is designed for maximum traceability and ease of expansion.

By using the capabilities of the FORTRAN95/2003 language, the new OOP architecture involves the use of pointers, lists and trees, data encapsulation, defined operators and assignments, operator extension and overloading, use of generic procedures, and maximum use of the powerful intrinsic vector and matrix processing operations (available in the extended mathematical library of FORTRAN 95/2003). This leads to increased computational efficiency, while allowing seamless applicability of the code to multi-processor parallel computing platforms. The result is a code that is

transparent and compact, and frees the developer from the tedium of tracking the disparate attributes that define the objects, thus enabling a quantum jump in the complexity of problem that can be tackled. This is demonstrated in **T+H**, in which 26 different phase combinations can be easily described. An additional feature of the FORTRAN 95/2003 language of TOUGH+ is the near complete interoperability with C/C++, which allows the interchangeable use of procedures written in either FORTRAN 95/2003 or C/C++, and makes possible the seamless interaction with pre- and post-processing graphical environments.

Based on insights provided by TOUGH2/EOSHYDR2 [*Moridis*, 2003], numerical 'bottlenecks' were removed, different (and fewer) primary variables were selected, and more powerful linearization techniques were employed in **T+H**, resulting in significant improvements in execution speed and numerical performance. Note that TOUGH+HYDRATE v1.2 still uses most of the inputs (and the input formats) used by the conventional TOUGH2 code to fulfill the functional requirement (part of the code design) of backward compatibility of the TOUGH+ family codes with older input data files used in TOUGH2 [*Pruess et al.,* 1999] simulations. More powerful input data file structures will be available in future releases of TOUGH+.

By solving the coupled equations of mass and heat balance, **T+H** can model the non-isothermal gas release, phase behavior and flow of fluids and heat in complex geologic media. The code can simulate production from natural CH_4-hydrate deposits in the subsurface (i.e., in the permafrost and in deep ocean sediments), as well as laboratory experiments of hydrate dissociation/formation in porous/fractured media. The only limitations on the size of the domain to be simulated are imposed by the underlying

physics. Thus, if the volume of the domain and its subdivision are such that (a) a representative volume can be defined and (b) Darcy's law applies, then **T+H** can be used for the prediction of the behavior of a hydrate-bearing geological system.

TOUGH+HYDRATE v1.2 includes both an equilibrium and a kinetic model of hydrate formation and dissociation. The model accounts for heat and up to four mass components, i.e., water, CH_4, hydrate, and water-soluble inhibitors such as salts or alcohols. These are partitioned among four possible phases (gas phase, liquid phase, ice phase and hydrate phase). Hydrate dissociation or formation, phase changes and the corresponding thermal effects are fully described, as are the effects of inhibitors. The model can describe all possible hydrate dissociation mechanisms, i.e., depressurization, thermal stimulation, salting-out effects and inhibitor-induced effects, both singly and in any combination.

Compared to TOUGH2/EOSHYDR2, **T+H** no longer tracks individually pre-existing free gas and gas from dissociation. While this was a convenient feature in TOUGH2/EOSHYDR2, it was not scientifically robust (because of the empirical and arbitrary differentiation between "native" and "hydrate-originating" gas), caused significant problems in complex simulations involving successive hydrate formation and dissociation (because of the difficulty of assigning gas provenance under these conditions), introduced an unacceptable convergence behavior (brought about by the empirical gas differentiation), increased the number of equations and introduced disproportionately large additional computational work (caused by the increased size of the problem and the demanding computational process for the estimation of binary gas

diffusion). Elimination of the gas tracking option in **T+H** greatly enhanced the convergence behavior, while reducing execution times by up to 50%.

While the TOUGH2/EOSHYDR2 capabilities to describe binary hydrates, and the properties and behavior of all hydrate forming gases are fully implemented in **T+H**, only methane hydrates can be simulated by this version of the code. The reason for this stems from the fact that our ability to mathematically describe the problem far exceeds the supporting fundamental knowledge on the subject. Treating the component hydrates as individual entities unaffected by each other or as segregated macroscopic quantities of hydrocarbon within the hydrate is incorrect because binary hydrates behave in a manner akin to solid solutions [*Sloan*, 1998]. Thus, changes in the composition of the hydrate and in the gas phase are functions not only of pressure and temperature, but also of concentration. Currently, there are no readily available P-T-X diagrams for such hydrates. The empirical distribution coefficient K_{vsi} method of *Carson and Katz* [1942] employed in TOUGH2/EOSHYDR2 was difficult, and often impossible, to converge (the highest convergence rate was 65%, and was observed in very simple problems), and lead to long execution times. The most physically and mathematically robust approach is to incorporate fast regression relationships based on the computationally intensive but conceptually sound statistical thermodynamics approach of *Sloan* [1998]. This option is currently being explored for incorporation into later version of the code if such a need arises.

Scoping simulations had indicated that sometimes fully 80% of the computational load in TOUGH2/EOSHYDR2 was due to the challenges of the distribution coefficient method. Given the fact that most of the most obvious hydrate deposit targets for gas

production involve almost exclusively pure CH_4-hydrates led to the decision to inactivate the binary hydrate capabilities in **T+H**.

This report provides a detailed presentation of the features and capabilities of **T+H**, and includes a thorough discussion of the underlying physical, thermodynamic and mathematical principles of the model in addition to the main governing equations. The various phase regimes and the corresponding primary variables are discussed in detail, as well as the reasons for their selection. Examples of input data files (and of sections of the corresponding output files) are included as an aide to the **T+H** user. Results from simulations of gas production from realistic hydrate-bearing geologic systems are also included.

2.0 Concepts, Underlying Physics, and Governing Equations

2.1. Modeled Processes and Underlying Assumptions

T+H can model the following processes and phenomena in hydrate-bearing geologic systems:

(1) The partitioning of the mass components among the possible phases

(2) The flow of gases and liquids in the geologic system

(3) The corresponding heat flow and transport

(4) Heat exchanges due to

   a. Conduction

   b. Advection/convection

   c. Radiation

   d. Hydrate reaction (dissociation or formation)

e. Latent heat related to phase changes (ice melting or water fusion, water evaporation or vapor condensation)

f. Gas dissolution

g. Inhibitor dissolution

(5) Equilibrium or kinetic hydration reaction (dissociation or formation),

(6) The transport of water-soluble gases and inhibitors (such as salts and alcohols), accounting for advection and molecular diffusion

(7) The effects of water-soluble inhibitors on the hydrate behavior

(8) Any method of hydrate dissociation (i.e., depressurization, thermal stimulation and inhibitor effects), and combinations thereof

A deliberate effort was made to keep the simplifying assumptions involved in the development of the underlying physical, thermodynamic and mathematical model to a minimum. These include:

(1) Darcy's law is valid in the simulated domain under the conditions of the study.

(2) In the transport of dissolved gases and inhibitors, mechanical dispersion is small compared to advection (by neglecting mechanical dispersion, memory requirements and execution times are substantially reduced).

(3) The compressibility and thermal expansivity of hydrate are the same as those of ice (necessitated by dearth of data on the subject).

(4) The movement of the geologic medium (soil heaving) while freezing is not described, and the effects on pressure (caused by density differences between the liquid and ice phases) are accommodated through a relatively high pore compressibility of the geologic medium.

(5) Dissolved salts do not precipitate as their concentration increases during water freezing. Consequently, the aqueous phase is not allowed to disappear when salts are present. This simplification was necessitated by (a) the marginal practical aspects of such a scenario, (b) the lack of fundamental knowledge and quantitative relationships describing the hydrate-salt interaction under these conditions, and (c) the computationally intensive requirements of describing the thermodynamics of dense brines and halite precipitation.

(6) The concentration of the dissolved inhibitors is such that it does not affect the thermophysical properties of the aqueous phase. Although the thermodynamics of inhibitor-water systems are well known and are already available within **T+H**, they are not invoked in this version (but will be considered as an option in future code releases) because their effect may not very important, they are computationally intensive, and inhibitor-induced dissociation is not considered an attractive first option for gas production from hydrate deposits.

(7) The inhibitor is non-volatile in the temperature-pressure range of the study. This simplification was introduced because of the limited practical possibility for such a scenario, and of the significant computational requirements to account for the inhibitor vapor pressure and diffusion in the gas phase.

(8) The pressure $P < 100$ MPa (6800 psi). The pressure-dependent equations describing the hydrate properties and behavior in **T+H** provide accurate solutions for a P over nine times larger than the largest pressure at which natural gas hydrates are known to exist (i.e., about 11 MPa). Thus, the existing capabilities can easily accommodate any natural or laboratory hydrate system.

Although equations for an accurate description of the thermophysical properties of hydrate systems for P as high as 1000 MPa are available in the code, this option is disabled because it involves an iterative process that increases the execution time by a factor of 3 or 4 even for $P < 100$ MPa.

2.2. Components and Phases

Depending on the thermodynamic state of the system, the amount of CH_4-hydrate created or CH_4 gas released is determined from the reaction

$$CH_4 + N_m H_2O = CH_4 \cdot N_m H_2O, \tag{2.1}$$

where N_H is the hydration number, and the subscript m denotes methane. In addition to simple CH_4-hydrates, natural hydrates may include one or more additional gasses. Such hydrate-forming gases exist practically in all natural hydrates and can play a significant role in their nucleation and behavior. The reaction describing the formation/hydration of a composite (binary) hydrate is

$$\chi_m \left[CH_4 \cdot N_m H_2O\right] + \chi_G \left[G \cdot N_G H_2O\right] = \chi_m CH_4 + \chi_G G + \left(\chi_m N_m + \chi_G N_G\right) H_2O, \tag{2.2}$$

where G is the second hydrate-forming gas, N is the hydration number, χ is the mole fraction in the binary hydrate, and the subscripts m and G denote the methane and the second gas, respectively. Obviously, $\chi_m + \chi_G = 1$. The gas G can be one of CO_2, H_2S, N_2, or another gaseous alkane $C_n H_{2n+2} (m = 2, 3, 4)$. In permafrost hydrate accumulations, χ_m routinely exceeds 95% [*Collett et al.*, 1999].

A non-isothermal hydrate system can be fully described by the appropriate mass balance equations and an energy balance equation. The following components κ (and the

corresponding indicators used in the subsequent equations), corresponding to the number of equations, are considered in **T+H**:

$\kappa \equiv$	h	hydrate
	w	water
	m	CH_4
	i	water-soluble inhibitor (salt or organic substance)
	θ	heat

These mass and energy components are partitioned among four possible phases β, which are listed below along with the corresponding indicators (used in the subsequent equations):

$\beta \equiv$	H	solid-hydrate (components: m,w - equilibrium, h - kinetic)
	A	aqueous (components: dissolved m, dissolved i)
	G	gaseous (components: m, vapor w)
	I	solid-ice (component: w)

Note that hydrate is both a component and a phase under kinetic hydration reaction conditions. Under equilibrium conditions, hydrate is treated only as a phase.

2.3. The Mass and Energy Balance Equation

Following *Pruess et al.* [1999], mass and heat balance considerations in every subdomain (gridblock) into which the simulation domain is been subdivided by the integral finite difference method dictates that

$$\frac{d}{dt}\int_{V_n} M^{\kappa}\, dV = \int_{\Gamma_n} \mathbf{F}^{\kappa}\cdot\mathbf{n}\, d\Gamma + \int_{V_n} q^{\kappa}\, dV\,, \tag{2.3}$$

where:

V, V_n volume, volume of subdomain n [L^3];

M^{κ} mass accumulation term of component κ [kg m^{-3}];

A, Γ_n surface area, surface area of subdomain n [L^2];

$\mathbf{F}^{\kappa}$ Darcy flux vector of component κ [kg m^{-2}s^{-1}];

$\mathbf{n}$ inward unit normal vector;

q^{κ} source/sink term of component κ [kg m^{-3}s^{-1}];

t time [T].

2.4. Mass Accumulation Terms

Under equilibrium conditions, the mass accumulation terms M^{κ} for the mass components in equation (2.3) are given by

$$M^{\kappa} = \sum_{\beta\equiv A,G,I} \varphi S_{\beta}\rho_{\beta}X_{\beta}^{\kappa}, \quad \kappa\equiv w,m,i \tag{2.4}$$

where

ϕ porosity [*dimensionless*];

ρ_{β} density of phase β [kg m^{-3}];

S_β saturation of phase β [*dimensionless*];

X_β^κ mass fraction of component $\kappa \equiv w, m, h$ in phase β [kg/kg]

In the equilibrium model, different cases of β give the following relations:

$$\beta \equiv A:\ X_A^h = 0$$
$$\beta \equiv G:\ X_G^h = X_G^i = 0$$
$$\beta \equiv H:\ X_H^w = \frac{W^m}{W^h},\ \ X_M^m = 1 - X_H^w,\ \ X_H^i = 0,\ \ X_H^h = 0$$
$$\beta \equiv I:\ X_I^m = X_I^h = X_I^i = 0,\ \ X_I^w = 1$$

The terms W^m and W^h denote the molecular weights of the CH_4 and of the hydrate, respectively. Thus, the values of X_H^w and X_M^m reflect the stoichiometry in Equation (2.1).

Under kinetic conditions, the mass accumulation terms M^κ in Equation (2.3) are given by

$$M^\kappa = \sum_{\beta \equiv A,G,H,I} \varphi S_\beta \rho_\beta X_\beta^\kappa,\ \ \kappa \equiv w, m, h, i \tag{2.5}$$

In the kinetic model, different cases of β give the following relations

$$\beta \equiv A:\ X_W^h = 0$$
$$\beta \equiv G:\ X_G^h = X_G^i = 0$$
$$\beta \equiv H:\ X_H^w = X_H^m = X_H^i = 0$$
$$\beta \equiv I:\ X_I^m = X_I^h = X_I^i = 0,\ \ \ X_I^w = 1$$

Under kinetic conditions, the equation describing the behavior of the hydrate mass component and phase is provided by the model of [*Kim et al.*, 1987] as

$$Q_H = \frac{\partial M}{\partial t} = -K_0 \exp\left(\frac{\Delta E_a}{RT}\right) F_A A \left(f_{eq} - f_v\right), \tag{2.6}$$

where

K_0 intrinsic hydration reaction constant [kg m^{-2} Pa^{-1} s^{-1}];

ΔE_a hydration activation energy [J mol^{-1}];

R universal gas constant [8.314 J mol^{-1} K^{-1}];

T temperature [^{o}K];

F_A area adjustment factor [dimensionless];

A surface area participating in the reaction [m^2];

f_{eq} fugacity at equilibrium at temperature T (Pa)

f_v fugacity in the gas phase at temperature T (Pa)

The surface area is computed by assigning the hydrate saturation uniformly to the interstitial spaces of the porous medium. To accomplish this, the original solid grain volume (considered to be composed of spherical particles) is determined as $V_p = \frac{4}{3}\pi r_p^3$, where r_p is the solid grain radius [m]. Then, the number of voids N_V (pore spaces) is assumed to be equal to the number of solid grains (a valid approach for spherical particles), and the corresponding void volume V_V is computed from

$$N_V = \frac{(1-\varphi)}{V_p}, \quad V_V = \frac{\varphi}{N_V}, \tag{2.7}$$

At the interface of pores and voids, the grain surface area is the same for both the grains and the voids, and is computed as $A_p = 4\pi r_p^2$, resulting in a total area (per unit volume) of $A_{TV} = N_V A_p$. Then the volume of the void is assumed to vary linearly with the r_V^3, where $r_V = 0.1547\ r_p$, is a representative radius describing the radius of the sphere fitting in the interstitial space between the spherical grains. Then, at any time t, a representative hydrate particle radius r_H and volume V_H are computed as

$$V_H = \frac{\phi S_H}{N_V}, \quad r_h = r_V\left(\frac{V_H}{V_V}\right)^{1/3} = r_V S_H^{1/3} \tag{2.8}$$

and the hydrate reactive area is computed as

$$A = f_A A_{TV} \left(\frac{r_H}{r_V} \right)^2 = f_A N_V \left(4\pi r_p^2 \right) S_H^{2/3}. \tag{2.9}$$

The area adjustment factor f_A accounts for the deviation of the interstitial volume from that based on the assumption of grain sphericity, and can incorporate heterogeneity effects related to the hydrate "particle" size and saturation distribution. Given the intrinsic permeability k of a porous medium, the Kozeny-Carman equation can provide an estimate of the average (effective) radius r_p of the porous medium grains as [*Bear*, 1972]

$$r_p = \left[45k \frac{(1-\phi)^2}{\phi^3} \right]^{1/2}$$

Alternatively, an estimate of r_p can be obtained from sieve analysis (if such data are available).

2.5. Heat Accumulation Terms

The heat accumulation term includes contributions from the rock matrix and all the phases, and, in the kinetic model, is given by the equation

$$M^\theta = (1-\phi)\rho_R C_R T + \sum_{\beta=A,G,H,I} \phi S_\beta \rho_\beta U_\beta + Q_{diss}, \tag{2.10}$$

where

$$Q_{diss} = \begin{cases} \Delta\left(\phi \rho_H S_H \Delta H^0\right) & \text{for equilibrium dissociation} \\ 0 & \text{for kinetic dissociation} \end{cases} \tag{2.11}$$

ρ_R rock density [kg m^{-3}];

C_R heat capacity of the dry rock [J kg^{-1} K^{-1}];

U_β specific internal energy of phase β [J kg^{-1}];

$\Delta()$ change in the quantity in parentheses over the current time step;

ΔU_H specific enthalpy of hydrate dissociation/formation [J kg^{-1}]

The specific internal energy of the gaseous phase is a very strong function of composition, is related to the specific enthalpy of the gas phase H_G, and is given by

$$U_G = \sum_{\kappa \equiv w,m} X_G^\kappa u_G^\kappa + U_{dep} \quad \left(= H_G - \frac{P}{\rho_G} \right), \tag{2.12}$$

where u_G^κ is the specific internal energy of component κ in the gaseous phase, and U_{dep} is the specific internal energy departure of the gas mixture [J kg^{-1}].

The internal energy of the aqueous phase accounts for the effects of gas and inhibitor solution, and is estimated from

$$U_A = X_A^w u_A^w + X_A^m \left(u_A^m + U_{sol}^m \right) + X_A^i \left(u_A^i + U_{sol}^i \right), \tag{2.13}$$

where u_A^w, u_A^m and u_A^i are the specific internal energies of H_2O, CH_4 and the inhibitor at the conditions prevailing in the aqueous phase, respectively, and U_{sol}^m and U_{sol}^i are the specific internal energies corresponding to the dissolution of CH_4 and of the inhibitor in water, respectively. The terms u_A^i and U_H are determined from

$$u_A^i = h_A^i - \frac{P}{\rho_i} = \int_{T_0}^{T} C_i dT - \frac{P}{\rho_i} \quad \text{and} \quad U_H = H_H - \frac{P}{\rho_H} = \int_{T_0}^{T} C_H dT - \frac{P}{\rho_H} \tag{2.14}$$

where T_0 is a reference temperature, h_A^i and H_H are the specific enthalpies of H_2O and hydrate (phase or component), respectively, and C_i and C_H are the temperature-dependent heat capacities of the inhibitor and the gas hydrate, respectively [J kg^{-1} K^{-1}].

2.6. Flux Terms

The mass fluxes of water, CH_4 and inhibitor include contributions from the aqueous and gaseous phases, i.e.,

$$\mathbf{F}^{\kappa} = \sum_{\kappa \equiv A,G} \mathbf{F}_{\beta}^{\kappa}, \quad \kappa \equiv w, m, i \tag{2.15}$$

The contributions of the two solid phases ($\beta \equiv H, I$) to the fluid fluxes are zero. Therefore, in the kinetic model the mass flux of the hydrate component ($\kappa \equiv h$) across all subdomain boundaries is

$$\mathbf{F}^{h} = 0 \tag{2.16}$$

For the aqueous phase, $\mathbf{F}_A^w = X_A^w \mathbf{F}_A$, and the phase flux $\mathbf{F}_A$ is described by Darcy's law as

$$\mathbf{F}_A = -k \frac{k_{rA} \rho_A}{\mu_A} \left(\nabla P_A - \rho_A \mathbf{g} \right), \tag{2.17}$$

where

k rock intrinsic permeability [m^2];

k_{rA} relative permeability of the aqueous phase [dimensionless];

μ_A viscosity of the aqueous phase [Pa s];

P_A pressure of the aqueous phase [Pa];

$\mathbf{g}$ gravitational acceleration vector [m s^{-2}].

The aqueous pressure P_A is given by

$$P_A = P_G + P_{cGW}, \tag{2.18}$$

where $P_G = P_G^m + P_G^w$ is the gas pressure [Pa], P_{cGW} is the gas-water capillary pressure [Pa], and P_G^m, P_G^w are the CH_4 and water vapor partial pressures [Pa] in the gas phase,

respectively. The CH_4 solubility in the aqueous phase is related to P_G^m through Henry's law,

$$P_G^m = H^m X_A^m, \tag{2.19}$$

where $H^m = H^m(P_G, T)$ [Pa] is a pressure- and temperature-dependent factor akin to Henry's constant. Note that it is possible to determine the CH_4 from the equality of fugacities in the aqueous and the gas phase. Although this approach provides a more accurate solution, the difference does not exceed 2-3% for the vast majority of CH_4-hydrate problems in reservoir or laboratory settings, but the execution time can increase as much as 30%.

The mass flux of the gaseous phase ($\beta \equiv G$) incorporates advection and diffusion contributions, and is given by

$$\mathbf{F}_G^\kappa = -k_0\left(1+\frac{b}{P_G}\right)\frac{k_{rG}\rho_G}{\mu_G}X_G^\kappa\left(\nabla P_G - \rho_G\mathbf{g}\right) + \mathbf{J}_G^\kappa, \quad \kappa \equiv w, m \tag{2.20}$$

where

k_0 absolute permeability at large gas pressures (= k) [m^2];

b *Klinkenberg* [1941] b-factor accounting for gas slippage effects [Pa];

k_{rG} relative permeability of the gaseous phase [*dimensionless*];

μ_G viscosity of the gaseous phase [Pa s].

The term $\mathbf{J}_G^\kappa$ is the diffusive mass flux of component κ in the gas phase [kg m^{-2} s^{-1}], and is described by

$$\mathbf{J}_G^\kappa = -\phi\underbrace{\left(\phi^{1/3}S_G^{10/3}\right)}_{\tau_0\tau_G} D_G^\kappa\rho_G\nabla X_G^\kappa = -\phi\left(\tau_0\tau_G\right) D_G^\kappa\rho_G\nabla X_G^\kappa, \quad \kappa \equiv w, m \tag{2.21}$$

where D_G^κ is the multicomponent molecular diffusion coefficient of component κ in the gas phase in the absence of a porous medium [m^2 s^{-1}], and τ_G is the gas tortuosity [dimensionless] computed from the *Millington and Quirk* [1961] model. The diffusive mass fluxes of the water vapor and CH_4 gas are related through the relationship of *Bird et al.* [1960]

$$\mathbf{J}_G^w + \mathbf{J}_G^m = 0, \tag{2.22}$$

which ensures that the total diffusive mass flux of the gas phase is zero with respect to the mass average velocity when summed over the two components ($\kappa \equiv w, m$). Then the total gas phase mass flux is the product of the Darcy velocity and density of the gas phase.

The flux of the dissolved inhibitor (EQN) is described by

$$\mathbf{F}_A^i = X_A^i \mathbf{F}_A + \mathbf{J}_W^i, \tag{2.23}$$

where

$$\mathbf{J}_W^i = -\varphi S_W \left(\varphi^{1/3} S_A^{7/3}\right) D_0 \rho_A \nabla X_A^i = -\varphi S_W \left(\tau_A\right) \; D_0 \rho_A \nabla X_A^i, \quad \kappa \equiv w, m \tag{2.24}$$

D_0 is the molecular diffusion coefficient of the inhibitor in water, and τ_A is the tortuosity of the aqueous phase.

The heat flux accounts for conduction, advection and radiative heat transfer, and is given by

$$\mathbf{F}^\theta = -\left[(1-\phi)K_R + \phi\left(S_H K_H + S_I K_I + S_A K_A + S_G K_G\right)\right]\nabla T + f_\sigma \sigma_0 \nabla T^4 + \sum_{\beta \equiv A,G} h_\beta \mathbf{F}_\beta, \tag{2.25}$$

where

K_R thermal conductivity of the rock [W m^{-1} K^{-1}];

K_β thermal conductivity of phase $\beta \equiv A, G, H, I$ [W m^{-1} K^{-1}];

h_β specific enthalpy of phase $\beta \equiv A, G, H, I$ [J kg^{-1}];

f_σ radiance emittance factor [dimensionless];

σ_0 Stefan-Boltzmann constant [5.6687×10^{-8} J m^{-2} K^{-4}].

The specific enthalpy of the gas phase is computed as

$$H_G = \sum_{\kappa \equiv w,m} X_G^\kappa h_G^\kappa + H_{dep}, \tag{2.26}$$

where h_G^κ is the specific enthalpy of component κ in the gaseous phase, and H_{dep} is the specific enthalpy departure of the gas mixture [J kg^{-1}]. The specific enthalpy of the aqueous phase is estimated from

$$H_W = X_A^w h_A^w + X_A^m \left(h_A^m + H_{sol}^m\right) + X_A^i \left(h_A^i + H_{sol}^i\right), \tag{2.27}$$

where h_A^w, h_A^m and h_A^i (see equation (2.14)) are the specific enthalpies of H_2O, CH_4 and the inhibitor at the conditions prevailing in the aqueous phase, respectively, and H_{sol}^m and H_{sol}^i are the specific enthalpy of dissolution [J kg^{-1}] of CH_4 and of the inhibitor in the aqueous phase, respectively.

2.7. Source and Sink Terms

In sinks with specified mass production rate, withdrawal of the mass component κ is described by

$$\hat{q}^\kappa = \sum_{\kappa \equiv A,G} X_\beta^\kappa q_\beta, \quad \kappa \equiv w, m \tag{2.28}$$

where q_β is the production rate of the phase β [kg m^{-3}]. For a prescribed production rate, the phase flow rates q_β are determined internally according to different options available

in **T+H**. For source terms (well injection), the addition of a mass component κ occurs at desired rates $\hat{q}^{\kappa}$ ($\kappa \equiv w, m$). Inhibitor injection can occur either as a rate as an individual mass component ($\hat{q}^{i}$) or as a fraction of the aqueous phase injection rate, i.e., $\hat{q}^{i} = X_A^i \hat{q}_A$, where X_A^i is the inhibitor mass fraction in the injection stream.

In the kinetic model, the additional sink/source terms corresponding to hydrate dissociation and release of CH_4 and H_2O must be accounted for. The source term for CH_4 thus becomes $\hat{q}^{m} = Q^{m}$, where the production rate Q^{m} [kg m^{-3} s^{-1}] of CH_4 is computed from Equation (2.6) as

$$Q^{m} = -\frac{W^{m}}{W^{c}} Q_H \ , \tag{2.29}$$

Similarly, the source term for water (liquid or ice) becomes $\hat{q}^{m} + Q^{m}$, where the hydrate-related release of water Q^{w} is determined from the stoichiometry of Equation (2.1) as

$$Q^{w} = -\frac{N_m W^{w}}{W^{c}} Q_H, \tag{2.30}$$

Under equilibrium conditions, the rate of heat removal or addition includes contributions of (a) the heat associated with fluid removal or addition, as well as (b) direct heat inputs or withdrawals (e.g., microwave heating), and is described by

$$\hat{q}^{\theta} \ = \ q_d + \sum_{\kappa=A,G} h_{\beta} q_{\beta} \tag{2.31}$$

Under kinetic conditions, the rate of heat removal or addition is determined from

$$\hat{q}^{\theta} \ = \ q_d + \sum_{\kappa=A,G} h_{\beta} q_{\beta} + Q_H \Delta H^{0}, \tag{2.32}$$

2.8. Thermophysical Properties

2.8.1. Water

The properties and parameters of liquid water and steam in **T+H** are provided by (a) fast regression equations based on data from *NIST* [2000] and (b) steam table equations developed by *IFC* [1967]. These equations are accurate up to 700 ^{o}C and 100 MPa. The code also incorporates additional capabilities extending the temperature and range to 3000 ^{o}C and 1000 MPa, but these are based on an iterative approach, are computationally very demanding, and, thus, are not invoked in the current version. The enthalpy, sublimation pressure and fusion/melting pressure of ice (on the ice-vapor and ice-liquid water equilibrium lines of the water phase diagram) are computed using fast regression equations from data obtained using *NIST* [2000]. Within the solid ice phase (to T = 50 K and P 200 MPa), ice densities were determined using the ice compressibility model of *Marion and Jakubowski* [2004] and the thermal expansivity data from *Dantle* [1962]. The ice enthalpy was computed using the heat capacity polynomial equation with the coefficients reported in *Yaws* [1999].

2.8.2. CH_4-Hydrate

The hydration number N_m, and the physical and thermal properties of the CH_4-hydrate are input functions of temperature in **T+H**. In the determination of CH_4-hydrate density as a function of pressure and temperature, the compressibility and thermal expansivity of ice are used in the absence of any data on the subject. The parameters of kinetic dissociation of the hydrate are also inputs to **T+H**. The hydration equilibrium pressure-

temperature relationship and the dependence of the heat of dissociation on temperature are discussed in Section 2.9.

2.8.3. CH_4 Gas

The properties of the gas phase are provided by the Peng-Robinson equation of state [*Peng and Robinson*, 1976], one of several options available in the real-gas property package (segment **`T_RealGas_EOS.f95`**, see Section 3.4) included in **T+H**. This package computes the compressibility, density, fugacity, specific enthalpy and internal energy (ideal and departure) of pure gases and gas mixtures over a very wide range of pressure and temperature conditions. Additionally, the package computes the gas viscosity and thermal conductivity using the method of *Chung et al.* [1988], and binary diffusivities from the method of *Fuller et al.* [1969] and *Riazi and Whitson* [1993].

The package also allows determination of gas solubility in water by equating fugacities in the gas and aqueous phases, which involves computation of activity coefficients (in the aqueous phase) and fugacities. However, scoping calculations indicated that, for pressures P<100 MPa, accurate estimates can be obtained by alternative correlation relationships included in **T+H**. Determination of gas solubility through fugacities and activity coefficients provides accurate estimates for pressures as high as 1000 MPa, but is also very computationally demanding, while providing little (if any) benefit for any pressure regime expected in CH_4-hydrate studies. Thus, this option is available but deactivated in the current code version.

2.9. Hydrate Phase Relationships

Of particular interest are the pressures and temperatures of the Lw-H-V and I-H-V three-phase lines in the H_2O-CH_4 diagram, which delineate the limits to hydrate formation/dissociation. The relationship between the equilibrium hydration pressure P_e and the corresponding equilibrium hydration temperature T_e in **T+H** can be obtained from two sources. The first is the regression equation of *Kamath* [1984]

$$P = \exp\left(e_1 + \frac{e_2}{T}\right), \tag{2.33}$$

where P is in KPa, T is in K,

$$e_1 = \begin{cases} 38.980 \\ 14.717 \end{cases}, \quad e_2 = \begin{cases} -8533.80 & \text{for} \quad 0°\text{C} > T_c \geq 25°\text{C} \\ -1886.79 & \text{for} \quad -25°\text{C} \geq T_c > 0°\text{C} \end{cases} \quad \text{and} \quad T = T_c + 273.15 \tag{2.34}$$

The second source is a general regression expression derived by *Moridis* [2003] based on data from several researchers reported by *Sloan* [1998]. The two relationships and their range are shown in **Figure 2.1**. Limited smoothing in the vicinity of the quadruple point Q_1 (Figure 2.2) was implemented to allow continuity of the derivatives and smooth phase changes. This is a necessity in the Newton-Raphson iterations implemented in TOUGH+ (see Section 3.4). Because of its limited range, and the discontinuity at the quadruple point, the parametric equations in **Figures 2.1** and **2.2** are the default option in **T+H**. Use of the *Kamath* [1984] equation is advised only when phase boundaries are known to not be crossed in the system.

There are limited data on the temperature dependence of heat of dissociation ΔH^0, which is often taken as constant over small temperature ranges. In **T+H**, ΔH^0 under

three-phase conditions (Lw-H-V and I-H-V) can be computed from either the simple equation of *Kamath* [1984] as

$$\Delta H^0 = C_1 + C_2 T, \tag{2.35}$$

where ΔH^0 is in cal/gmol, T is in K,

$$C_1 = \begin{cases} 13,521 \\ 6,534 \end{cases}, \quad \text{and} \quad C_2 = \begin{cases} -4.02 & \text{for} \quad 0\ ^\circ\text{C} > T_c \geq 25\ ^\circ\text{C} \\ -11.97 & \text{for} \quad -25\ ^\circ\text{C} \geq T_c > 0\ ^\circ\text{C} \end{cases} \tag{2.36}$$

Alternatively, it can be determined from the Clausius-Clapeyron equation

$$\Delta H^0 = ZRT^2 \frac{d \ln P}{dT}, \tag{2.37}$$

using the gas compressibility Z and the known $d(\ln P)/dT$ of the hydrate equilibrium relationship (from the regression curve, see **Figures 2.1** and **2.2**). **Figure 2.3** shows the uncorrected ΔH^0 relationships from both equations. For comparison, **Figure 2.3** includes the laboratory measurements of ΔH^0 reported in *Sloan* [1998]. **Figure 2.4** shows the modified relationship used in **T+H**, which incorporates density-change corrections to the Clausius-Clapeyron equation, as well as smoothing in the immediate vicinity of the quadruple point.

There are no specific measurements of the equilibrium P -T relationship along the I-H-L_w and the I-V-L_w phase lines of a H_2O-CH_4O system, but is generally considered to follow the solidus line (melting/fusion equilibrium) of the water-ice system [*NIST*, 2000]. Thus, the equilibrium P-T relationship along the I-Lw-H phase line is computed as

$$P = P_Q - 6.26 \times 10^5\ (1.0 - T_d^{-3}) + 1.97135 \times 10^5 (1.0 - T_d^{21}) \tag{2.38}$$

where P is in Pa, T_d = T/273.16 (T in K), P_Q (in Pa) is the pressure at the hydrate quadruple point (see **Figures 2.1** and **2.2**). Finally, temperature was considered invariable

(and equal to 0 ^{o}C along the I-V-L_w phase line). The complete phase diagram of the water--CH_4--hydrate system is shown in **Figure 2.4**.

2.10. Inhibitor Effects on Hydrate Equilibrium

The effect of salinity on the dissociation equilibrium pressure-temperature relationship is described by the equation of *Dickens and Quinby-Hunt* [1997]

$$T_s = \left[\frac{1}{T_H} - \frac{n_m \Delta H^f}{\Delta H} \left(\frac{1}{273.15} - \frac{1}{T_f} \right) \right]^{-1} \tag{2.39}$$

In this equation, T_s is the equilibrium dissociation temperature in the salt solution (K), ΔH^f is the heat of fusion of ice (J/kg), T_H is the equilibrium dissociation temperature in the presence of pure water (K), and T_f is the freezing point of the salt solution (K). The predictions of this equation are in broad agreement with the measurements of Wright et al. (1999). Further analysis allowed the above complex equation to be replaced with the simpler and equally accurate relationship, which has the additional benefits of applying to both salts and inhibitors such as alcohols. In the simplified equation, the temperature depression (shift) induced by the inhibitors is computed as

$$\Delta T_D = \Delta T_{D,r} \frac{\ln\left(1 - x_A^i\right)}{\ln\left(1 - x_{A,r}^i\right)}, \tag{2.40}$$

where

x_A^i mole fraction of the inhibitor in the aqueous phase;

$x_{A,r}^i$ reference mole fraction of the inhibitor in the aqueous phase [K];

ΔT_D inhibitor-induced temperature depression [K];

$\Delta T_{D,r}$ temperature depression at the reference mole fraction $x^{i}_{A,r}$.

This approach is entirely consistent with the equation of *Makogon* [1981] for alcohols. Exploratory calculations have shown this equation to be within 3-5% of the estimates of the far more computationally intensive method of *Dickens and Quinby-Hunt* [1997].

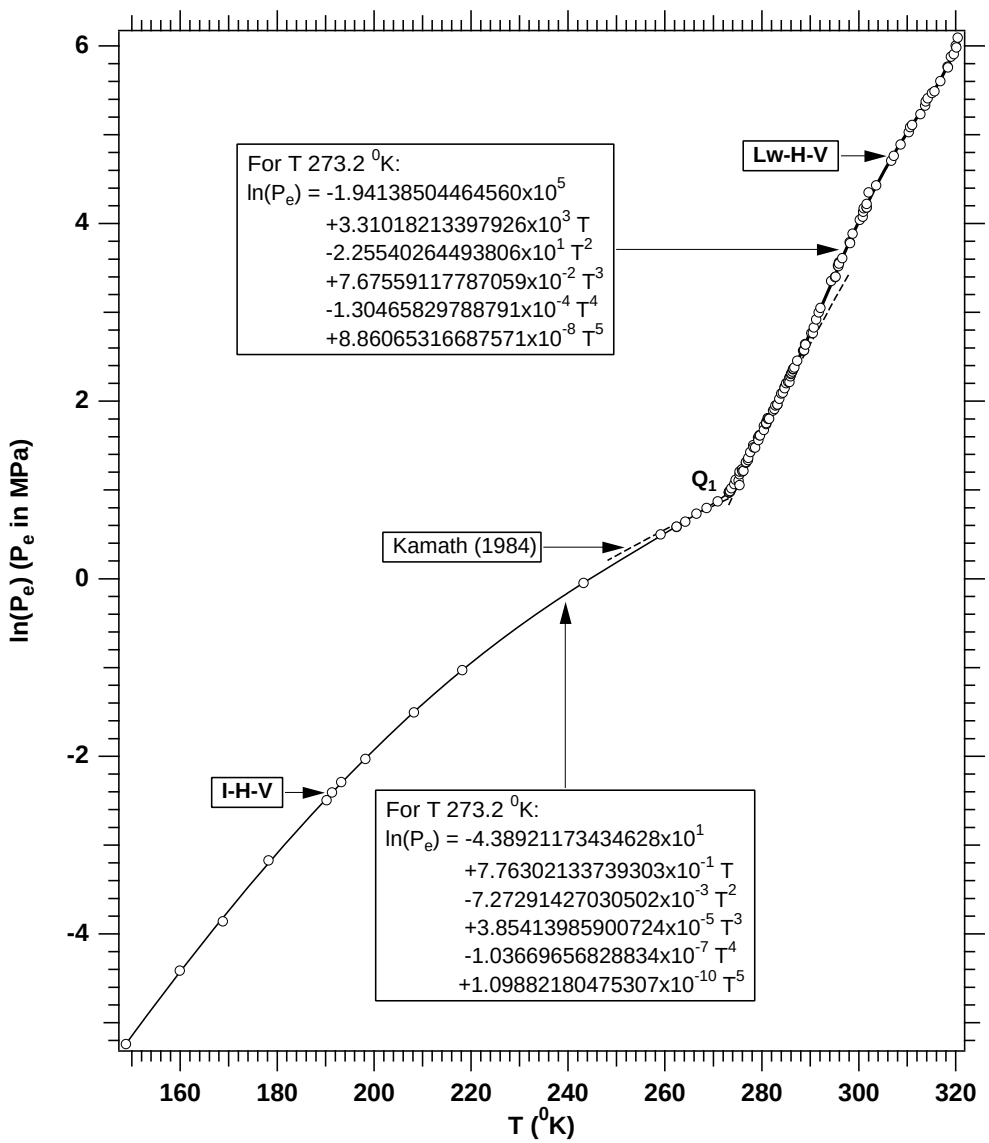

Figure 2.1. Relationship of the equilibrium hydration pressure *Pe* and temperature *Te* of the CH_4-hydrate [*Moridis,* 2003].

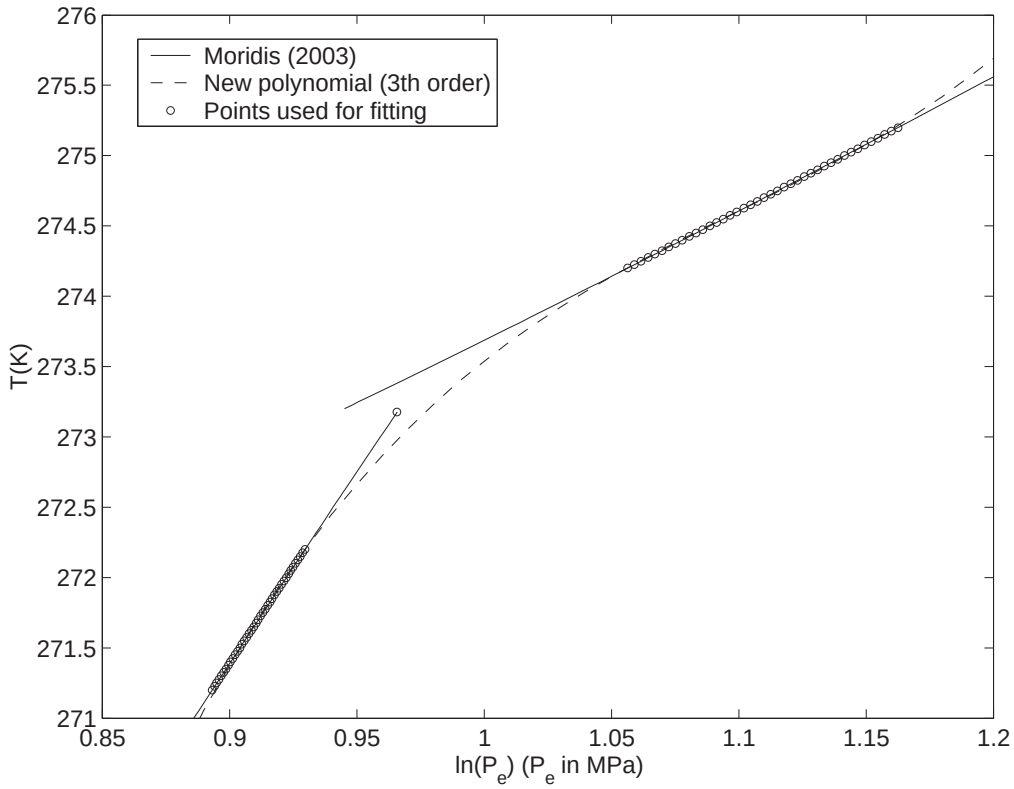

Figure 2.2. Modified *Pe -Te* relationship in the vicinity of the quadruple point.

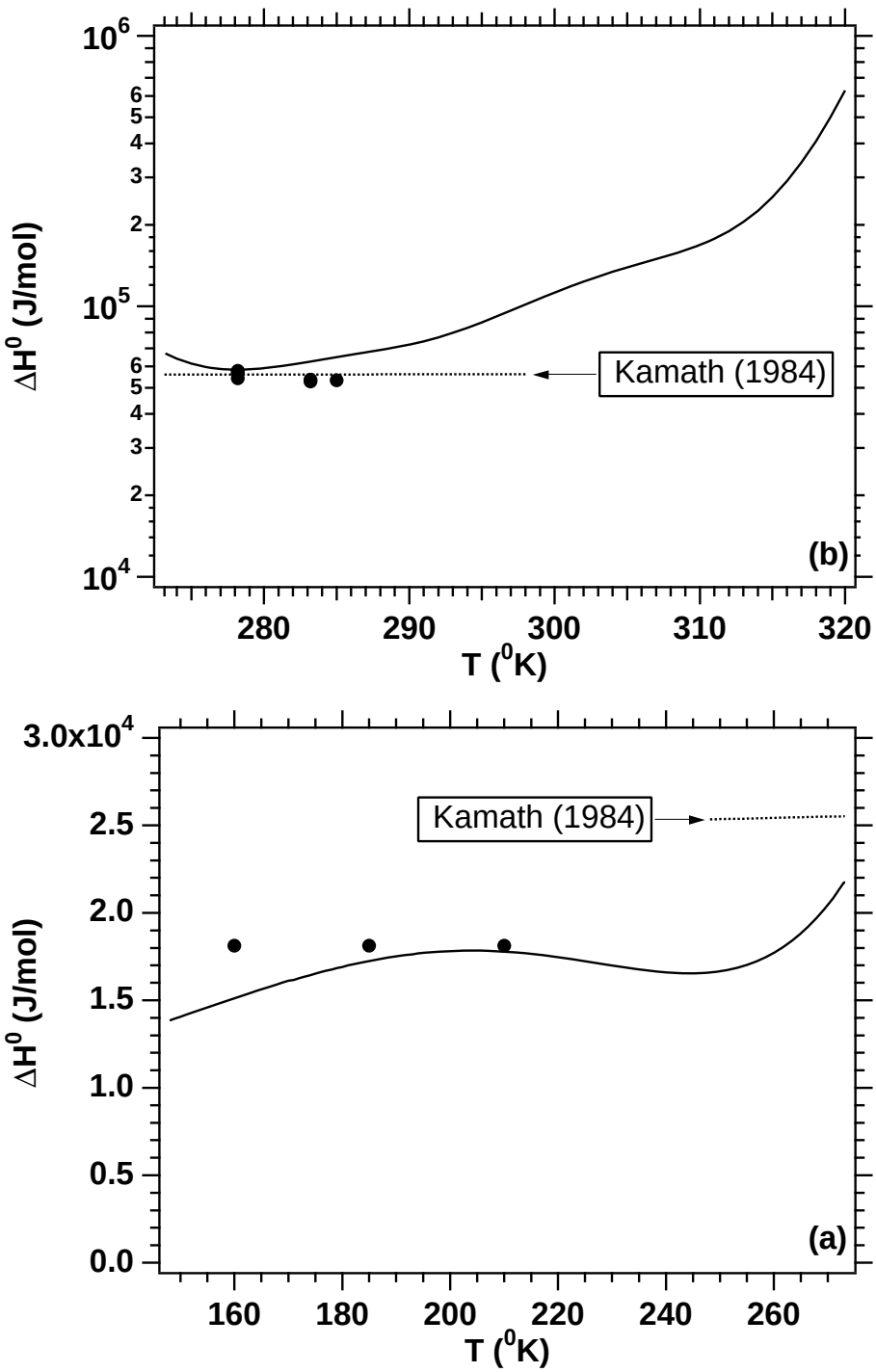

Figure 2.3. Temperature dependence of the CH_4-hydrate heat of dissociation ΔH^0 along the (a) I-H-V and (b) L_w -H-V phase lines [*Moridis*, 2003].

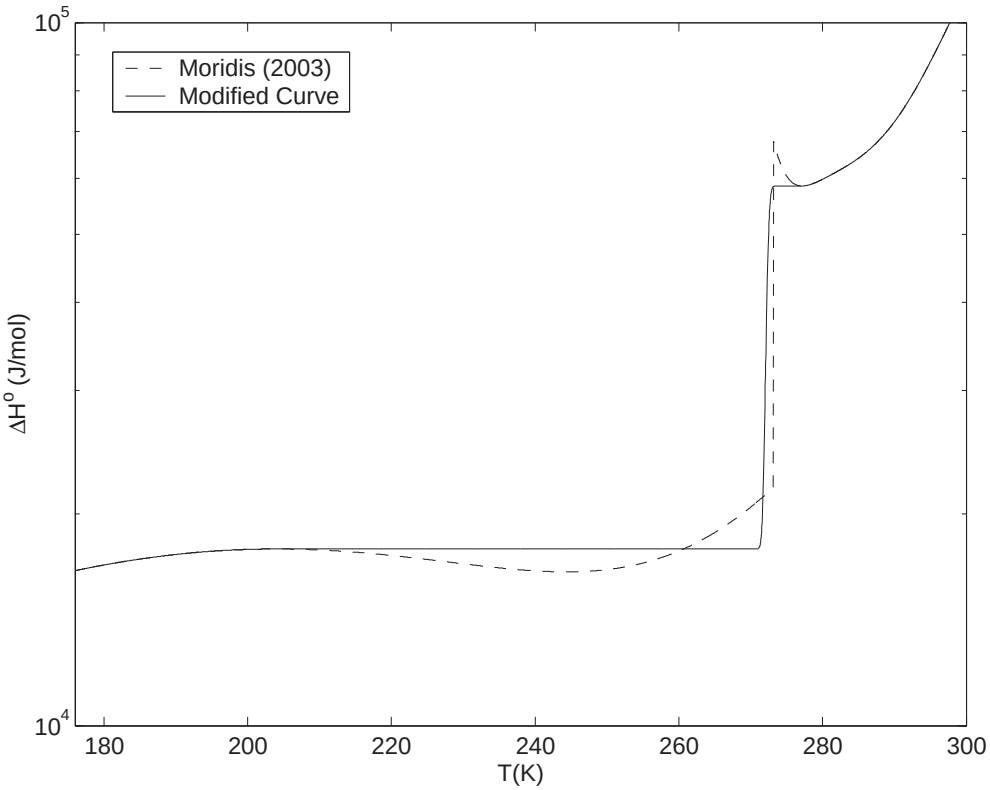

Figure 2.4. Modified ΔH^0 vs. T*e* relationship used in TOUGH+HYDRATE.

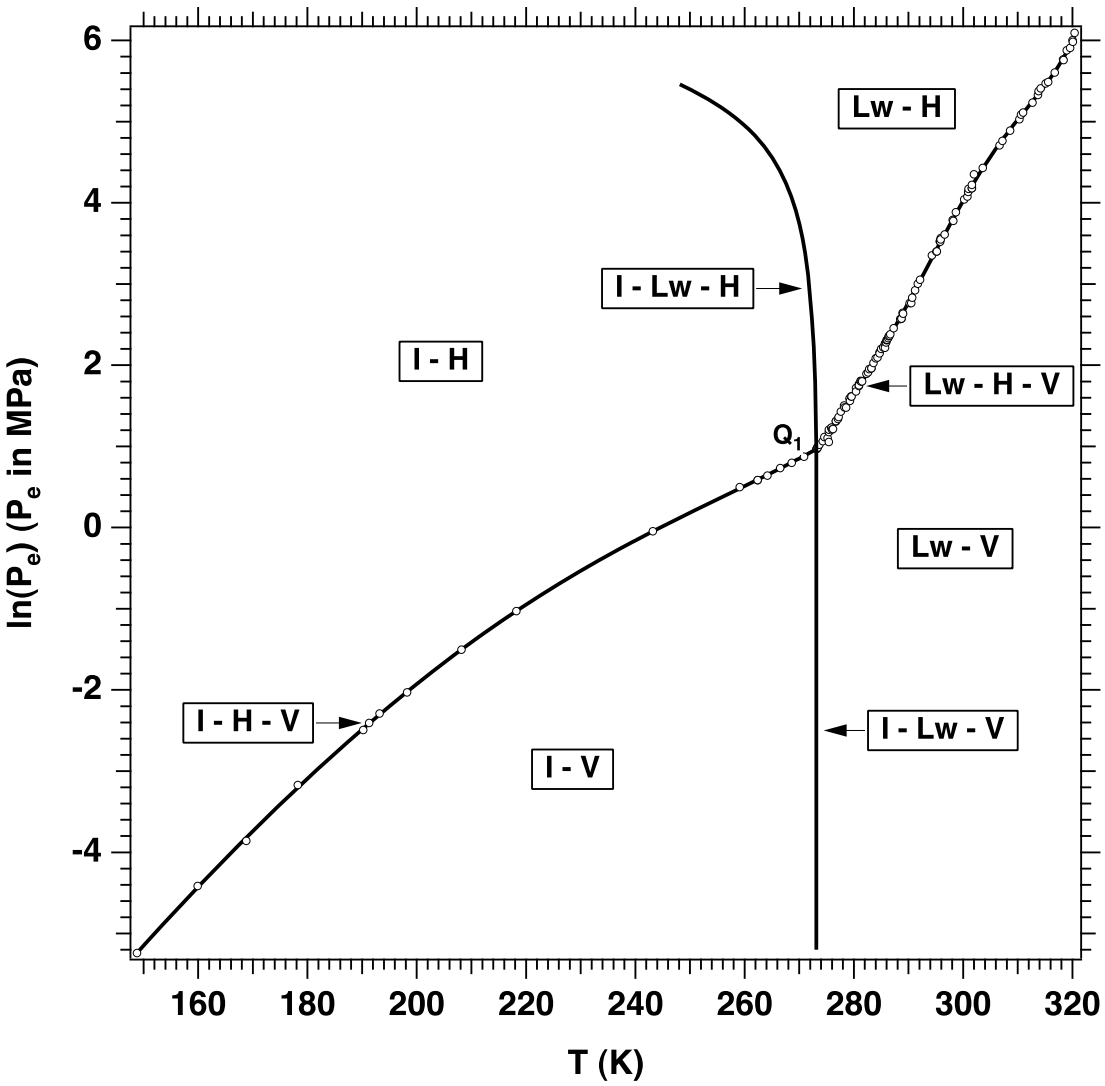

Figure 2.5. Pressure-temperature equilibrium relationship in the phase diagram of the water--CH4--hydrate system in T+H.

2.11. Porosity and intrinsic permeability change

When solid phases are deposited in a porous medium, through chemical precipitation or freezing of pore fluids, the ability of the porous medium to transmit fluids can change profoundly. The deposition of solids in a porous medium reduces the void space available for fluids. Such reduction in porosity will give rise to a reduction in permeability as well. There is an extensive literature, going back to the 1920s, about the manner in which permeability declines as portions of the pore space are filled by solids, and a bewildering variety of porosity-permeability correlations have been obtained from experimental and theoretical studies [*Scheidegger*, 1974, and references therein; *Morrow et al*., 1981; *Vaughan*, 1987; *Verma and Pruess*, 1988; *Phillips*, 1991; *Pape et al*., 1999; *Xu et al*., 2004]. Within the scope of the work undertaken here it is not possible to perform a thorough review of different permeability reduction models, and to evaluate their suitability for representing permeability reduction due to formation of hydrate and/or ice. Dearth of relevant information prevents considering whether any of the models developed for solid precipitation in porous media are valid to gas/water/ice/hydrate systems. Instead, we briefly discuss salient features of pore channels to highlight the most important effects, and then explain the rationale behind the preliminary choices made in this study.

It is obvious that permeability effects depend not just on the magnitude of porosity change, but on geometric properties of the pore channels, and on where and how solid deposition in those channels occurs. The lack of unanimity among different investigators about the correlation between porosity and permeability change reflects the great diversity of pore channel geometries and precipitation processes in porous media. The

simplest models conceptualize porous media as bundles of capillary tubes, which gives rise to a simple power law dependence of permeability k on porosity ϕ,

$$\frac{k}{k_0} = F_{\phi S} = \left(\frac{\phi}{\phi_0}\right)^n \tag{2.41}$$

where the term $F_{\phi S}$ is a *permeability adjustment factor* that describes the effects of (a) the presence of solid phases other than the medium grains (such as ice, hydrate or precipitating salts), and (b) changes in porosity on permeability, and the subscript "00" denotes properties at a reference state.

The exponent n typically will be in the range from 2 to 3 [*Phillips*, 1991], giving a rather mild dependence of permeability on porosity. However, in media with intergranular porosity, pore channels generally have a convergent-divergent geometry, consisting of a succession of '*pore throats*' with small radius and '*pore bodies*' with large radius (**Figure 2.6**).

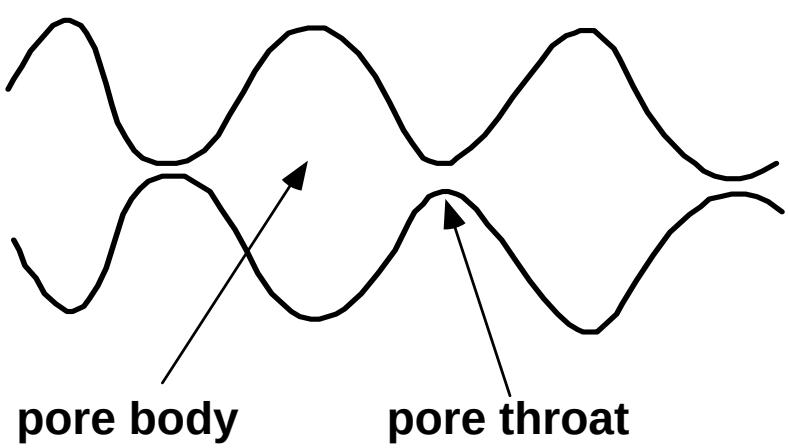

Figure 2.6. Schematic of pore channels, showing convergent-divergent geometry with a succession of pore throats and pore bodies.

If solids are deposited uniformly along the pore walls, or are preferentially deposited in the throats, then even relatively minor amounts of deposition can give rise to a dramatic decrease in permeability. Such behavior has been observed in field and laboratory-scale systems, including the diagenesis of sandstones [*Pape et al*., 1999], precipitation around geothermal injection wells [*Xu et al*., 2004], and hydrothermal flows in laboratory specimen [*Morrow et al*., 1981; *Vaughan*, 1987].

In these systems, a rather modest amount of precipitate, that leaves most of the original pore space available for fluids, nonetheless caused order-of-magnitude changes in absolute permeability. Such behavior can be understood from '*tubes-in-series*' models of pore space, as shown in **Figure 2.7** [*Verma and Pruess*, 1988]. If one assumes that solids are deposited as a layer of uniform thickness on the pore walls, then permeability will be reduced to zero when the throats become clogged, while plenty of (disconnected) porosity remains in the pore bodies. This leads to the concept of a non-zero '*critical porosity*' ϕ_c at which permeability is reduced to zero, with a permeability reduction as given in Equation (2.41) [*Verma and Pruess*, 1988; *Xu et al*., 2004].

$$\frac{k}{k_0} = F_{\phi S} = \left(\frac{\phi - \phi_c}{\phi_0 - \phi_c} \right)^n \tag{2.42}$$

Fractal models also give a very strong dependence of permeability on porosity, with exponents in relationships such as Equation (2.41) as large as $n = 10$ or more [*Pape et al*., 1999]. The above discussion clearly indicates the need fundamental research for the determination of the sites within the porous media at which hydrates and ice form preferentially.

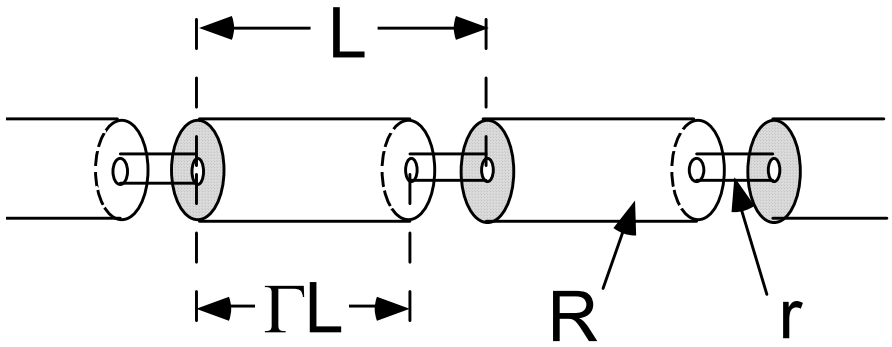

Figure 2.7. Tubes-in-series model of pore channels.

2.12. Wettability Phenomena

2.12.1. Capillary pressure

Effects of solids deposition become considerably more complicated, and involve more than just permeability change, when multiple fluid phase are present, such as water and gas. Surface tension effects between different phases give rise to '*capillary pressures*', denoted by P_{cap}, which may be profoundly altered when solids are deposited. Most mineral surfaces are preferentially wetted by water and, under partially saturated conditions, the pressure of the wetting (aqueous) phase inside a porous medium will be less than that of the non-wetting (gas) phase. The pressure difference $P_{cap} = P_g - P_l$ is termed capillary pressure, because it relates to the phenomenon of water level rise in a capillary tube. (Sometimes P_{cap} is defined with the opposite sign.)

Issues relating to capillary pressure can be conveniently discussed with reference to the pore size distribution of the porous medium. **Figure 2.8** shows a schematic probability density function (pdf) for pore sizes, which expresses the probability *p(r)* of

having pores with radius r. At a given capillary pressure P_{cap}, pores with radius $r' \leq r$ may be water-filled, where the cutoff radius r is related to the capillary pressure by the Young-Laplace equation (reference)

$$P_{cap} = \frac{2\sigma\cos(\gamma)}{r} \tag{2.43}$$

Here, σ is the surface tension (energy per unit surface area) at the water-gas interface, and γ is the contact angle, which usually is close to zero for preferentially water-wet minerals.

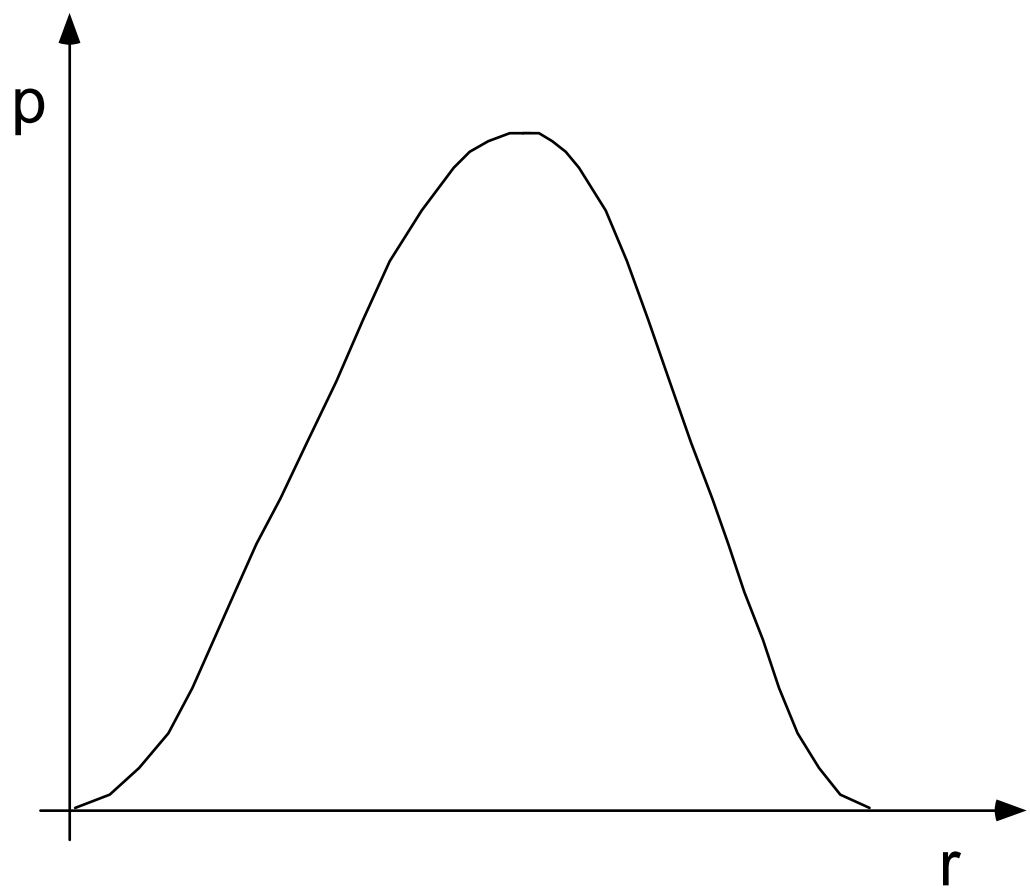

Figure 2.8. Schematic of probability density function p(r) for pore size distribution.

In introducing Equation (2.41) we stated that pores with $r' \leq$ r may be water filled, but whether indeed all pores with $r' \leq r$ will be water-filled at a prevailing capillary pressure P_{cap} given Equation (2.41) (and those with $r' > r$ will all be gas-filled) depends upon issues of *pore accessibility* that are not captured by the pore size distribution. The issue of accessibility arises because pores of certain radius may be entirely surrounded by larger pores, or by smaller pores, so that either water may not enter them during a wetting process, or may not be removed from them during a draining process. Accessibility gives rise to the well-known phenomenon of *capillary hysteresis*, where at a given magnitude of capillary pressure, water saturation will generally be larger during a drainage process than during a wetting process [*de Marsily*, 1986]. Although capillary hysteresis is a well-established effect, it is seldom taken into account in modeling applications; this is partially due to numerical difficulties associated with it, partially because information on applicable parameters is rarely available.

Formation of solid phases will alter the pore size distribution, generally reducing pore sizes, and thereby giving rise to stronger capillary pressures (see Equation 2.41). In order to estimate these changes, we require information on the original pore size distribution of the medium, and on the manner in which the pore size distributions will be altered during solids deposition. As no such information is presently available for the medium studied here, we proceed in a more phenomenological manner, and relate changes in capillary pressures to overall changes in porosity and permeability of the medium. Examining a variety of unconsolidated media, *Leverett* [1941] determined a dependence of capillary pressure on permeability and porosity, as follows.

$$P_{cap}(S_A) \quad = \quad \sqrt{\frac{k_{00}}{k} \cdot \frac{\phi}{\phi_{00}}}\, P_{cap,00} \tag{2.44}$$

where $P_{cap,00}$ is the capillary pressure corresponding to **a reference medium at the reference conditions**, at which the permeability and porosity of the porous medium are k_{00} and ϕ_{00}, respectively. Equation (2.44) is used in the present analysis, in the following manner. We represent *active* solids (i.e. ice that may melt or hydrate that may dissociate, as opposed to solid minerals which are inert) by means of a *solid saturation*, denoted by $S_S = S_H + S_I$, which measures the fraction of active pore space occupied by solids. The fraction of pore space available to fluid phases is $S_A + S_G$, and we have the constraint

$$S_A + S_G \quad = \quad 1 - S_S \tag{2.45}$$

Let $P_{cap,00}$ denote the capillary pressure function applicable to a porous medium free of solid saturation ($S_S = 0$) with a reference porosity ϕ_{00}. The total current porosity ϕ and the active (or available) porosity ϕ_a available to fluids are then defined by the equation

$$\phi_a = \underbrace{\left(F_{PT}\ \phi_{rr}\ \phi_{00}\right)}_{\phi}\ (S_A + S_G)\ \Rightarrow \frac{\phi_a}{\phi_{00}} = F_{PT}\ \phi_{rr}\ (S_A + S_G) = F_{PT}\ \phi_{rr}\ \left(1\text{-}S_S\right)\ , \tag{2.46}$$

where the term F_{PT} is a *porosity adjustment factor* that accounts for the effects of pressure and temperature on porosity, and ϕ_{rr} is the ϕ *relative magnitude*, which relates the porosity ϕ_0 of a given medium to the porosity ϕ_{00} of the reference medium at the same reference P and T. This term is introduced to account for situations in which the reference medium is different from the one under consideration, as is often the case when insufficient data are available and parameter estimation is based on scaling using known media as references. For a reference medium different from the one under consideration, $\phi_{rr} = \phi_0/\phi_{00}$. It is obvious that $\phi_{rr} = 1$ when the same medium is used as reference.

When changes in P and T are not large in geomechanically stable media, then F_{PT} can be estimated from the following equation:

$$F_{PT} = \frac{\phi}{\phi_{rr}\ \phi_{00}} = \frac{\phi}{\phi_0} = \exp\left[\alpha_P\ \Delta P + \alpha_T\ \Delta T\right] \approx 1 + \alpha_P\ \Delta P + \alpha_T\ \Delta T, \quad (2.47)$$

where $\Delta P = P - P_0$, $\Delta T = T - T_0$, α_P and α_T are the pore compressibility and thermal expansivity, respectively (see discussion in Section 6.2). For large ΔP and/or ΔT in compressible or geomechanically unstable media, F_{PT} can be estimated from a full geomechanical model that relates the resulting changes in geomechanical stresses and strains to changes in porosity.

The argument in the capillary pressure function $P_{cap,0}$ on the r.h.s of Equation (2.44) is the aqueous saturation S_A, referred to total fluid porosity. We measure liquid saturation on a scale that refers to total *active* (fluid plus hydrate- and ice-filled) pore space in the ice- and hydrate-free porous medium. In the medium with solid saturation S_S, S_A corresponds to a *scaled* saturation

$$S_A^* = \frac{S_A}{S_A + S_G} \quad (2.48)$$

relative to fluid-filled pore space, and this is the value to be used in the estimation of $P_{cap,0}$ in the r.h.s of Equation (2.44). In the next session we will discuss the estimation of $k_{r\phi}$ in the presence of solid phases (i.e., ice and/or hydrate) for use in Equation (2.44).

2.12.2. Relative Permeability

In multiphase flow, each fluid phase occupies only part of the pore space, and its effective permeability is reduced due to interference with the other phase(s). This effect is

represented by means of *permeability reduction factors* or *relative permeabilities*, customarily denoted by k_{rA} and k_{rG} for liquid (aqueous) and gas, respectively, such that effective permeability k_β to phase β (= A, G) is given by

$$k_\beta \;=\; k\, k_{r\beta}, \quad \text{where} \quad k \;=\; k_0\, F_{\phi S} \;=\; k_{00}\, k_{rr}\, F_{\phi S}, \tag{2.49}$$

and k_{rr} is the *k relative magnitude* that relates the permeability k_0 of a given medium to k_{00} of the reference medium at the same P and T. As in the case of ϕ_{rr}, k_{rr} is introduced to account for situations in which the reference medium is different from the one under consideration, as is often the case when insufficient data are available and parameter estimation is based on scaling such as the one described by Equation (2.44) using known media as references. For a reference medium different from the one under consideration, $k_{r\square} = k_0/k_{00}$. It is obvious that $k_{r\square} = 1$ when the same medium is used as reference.

The relative permeabilities are functions of the phase saturations S_β (fraction of pore space occupied by phase β), $k_{r\beta} = k_{r\beta}(S_\beta)$, and are usually obtained by measurement on laboratory specimen of porous media. **Figure 2.9** gives examples of commonly used liquid and gas relative permeabilities.

The partitioning of effective permeability to a fluid phase β into a porous medium- and solid-saturation dependent part ($k_0\, F_{\phi S}$) and a fluid saturation-dependent part $k_{r\beta}$ is a matter of convention and convenience. It leads to a conceptual ambiguity in the representation of permeability reduction from solid deposition in multiphase flow. Indeed, such permeability reduction may be attributed either to a change in *absolute* or *intrinsic* permeability (as described by the product $k_0\, F_{\phi S}$), as is done for single-phase flow, or it may be attributed to a change in the fluid relative permeability $k_{r\beta}$.

When hydrate and/or ice forms inside a partially water-saturated porous medium, such formation clearly must start in the water-filled portion of the pore space, but may not remain limited to the water-filled portion, as the solid crystals may grow and extrude into primarily gas-filled pores. In the absence of specific pore-scale information about where hydrate and/or ice will likely form, it is not possible to ascertain the applicability of relationships such as Equation (2.41) to the permeability reduction associated with hydrate formation and/or freezing. Even if applicable, appropriate parameters for the problem under study here are lacking.

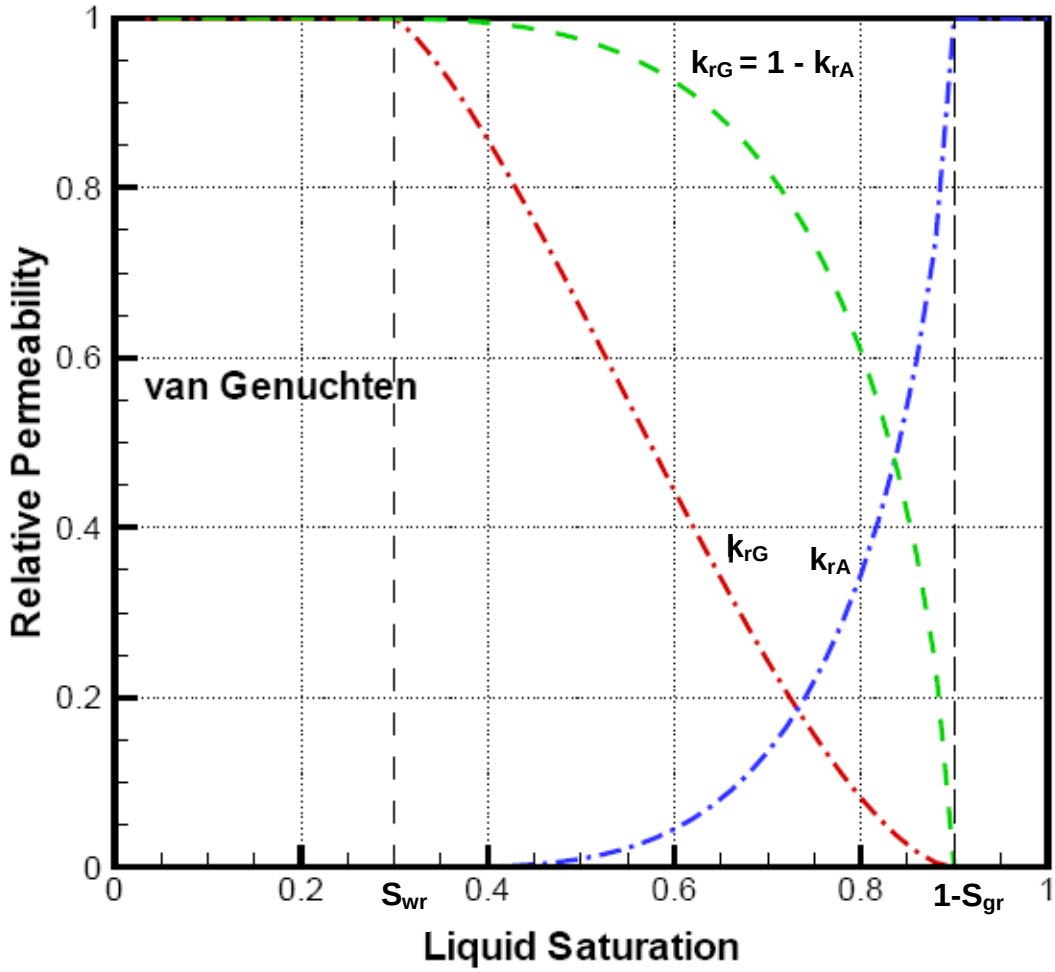

Figure 2.9. Liquid and gas relative permeabilities based on the van Genuchten model [*Finsterle*, 1999].

We therefore propose two alternative models to describe the wettability processes (relative permeability and capillary pressure) in hydrate- and/or ice-bearing media. The first model, hereafter referred to as the "*Original Porous Medium*" (OPM) model, is based on the treatment of (a) the medium porosity as unaffected by the emergence of hydrates and/or ice (although subject to change due to changes in pressure and temperature), (b) of the intrinsic permeability of the porous media as unchanging during the evolution of the solid phases, and (c) of the fluid flow as a relative permeability issue controlled by the saturations of the various phases in the pores. The second family of

models, hereafter referred to as the "*Evolving Porous Medium*" (EPM) models, consider the evolution of the solid phases (hydrate and ice) as tantamount to the creation of a new porous medium with continuously changing porosity and intrinsic permeability, the pore space of which is occupied only by the two fluid phases (aqueous and gas).

2.12.2.1. The OPM model. This simpler model represents permeability reduction as relative permeability effects, and does not require any new parameters to be introduced. More specifically, this model assumes that in the presence of solid phase(s), relative permeability to each fluid phase is given by the same function $k_{r\beta}(S_\beta)$ as in the absence of solids. This means that aqueous phase relative permeability $k_{rA} = k_{rA}(S_A)$ depends only on the aqueous saturation S_A, and is the same, regardless of how the remaining fraction $(1\text{-}S_A)$ of the pore space is divided between gas and solid phases. A similar comment applies to gas relative permeability $k_{rG} = k_{rG}(S_G)$.

Setting aside for a moment the issue of $(k_{r\phi}\ k_{rS})$, permeability reduction for the fluid phases then occurs simply because, when S_S increases, fluid phase saturations S_A and S_G generally must decrease also, as dictated by the constraint in Equation (2.43). This prescription is tantamount to asserting that liquid phase flow behaves as though solids deposition occurs entirely in what would otherwise be gas-filled pore space, while gas phase flow behaves as though solids deposition occurs entirely in what otherwise would be liquid-filled pore spaces. It is obvious that solids deposition cannot simultaneously occur only in liquid and only in gas-filled pore spaces, which points to a limitation of the proposed permeability reduction model. We nonetheless feel that a model that introduces no new and uncertain parameters is preferable to a model that does.

The permeability adjustment factor is computed from the following expression:

$$F_{\phi S} = k_{r\phi}\, k_{rS}, \tag{2.50}$$

where $k_{r\phi}$ is the *permeability ϕ-factor* that describes the effect of changes in ϕ on permeability, and k_{rS} is the *permeability S-factor* that relates reduction in the intrinsic permeability to the presence of solid phases (such as ice, hydrates or precipitating salts).

In the OPM model, $k_{rS} = 1$ by definition, and the permeability ϕ-factor in Equation (2.49) can be computed as

$$k_{r\phi} = \begin{cases} 1 & \text{when the effect of } \phi \text{ changes on } k \text{ is neglected} \\ \exp\left[\gamma\left(F_{PT} - 1\right)\right] & \text{when the effect of } \phi \text{ changes on } k \text{ is accounted for,} \end{cases} \tag{2.51}$$

where γ is an empirical parameter [*Rutqvist and Tsang*, 2002], and F_{PT} is computed from Equation (2.47).

In the OPM model, capillary pressures are estimated from Equation (2.44), in which:

- $\phi/\phi_{00} = \phi_{rr} F_{PT}$ is computed from Equation (2.47)
- $k_{00}/k = 1/k_{rr}\, k_{r\phi}$ is computed from Equations (2.49) and (2.50), and
- $P_{cap,00}$ is computed based on the scaled saturations S^* of Equation (2.48)

Thus, the final expression for estimating the capillary pressure in the OPM model is:

$$P_{cap} = \sqrt{\frac{\phi_{rr}}{k_{rr}} \cdot \frac{F_{PT}}{k_{r\phi}}} \cdot P_{cap,00}(S^*) \tag{2.52}$$

Additional scaling can be introduced by using the active porosity ϕ_a and ϕ_a/ϕ_{00} from Equation (2.46) – as opposed to ϕ/ϕ_{00} from Equation (2.47) – in the computation of Equation (2.44).

2.12.2.2. The EPM models. In recognition of, and attempting to overcome, the limitations of the OPM permeability reduction model, we have proposed two EPM models and performed sensitivity studies using the absolute (intrinsic) permeability modifications that will be discussed in this section. While the EPM models provide valuable insights, a more consistent and defensible model – based on both theoretical analyses and laboratory and field studies – for the effects of emerging solid phases on fluid permeabilities should be developed in the future.

EPM Model #1. With intrinsic permeability modification based on relative permeabilities, what absolute permeability should be used in the Leverett scaling Equation (2.41)? An attractive possibility would be to set $k = k_0\ (k_{rA} + k_{rG})$. This, however, is not acceptable because the sum of liquid and gas relative permeabilities depends not just on the solid saturation (S_S), but on S_A and S_G individually.

As a plausible alternative, we consider the permeability reduction when the fluid-available pore space is either entirely liquid-filled or entirely gas-filled, which leads to

$$k_{rS} = k_{rA}\left(S_A = 1 - S_S\right) \quad \text{or} \quad k_{rS} = k_{rG}\left(S_G = 1 - S_S\right).$$

Note that either expression depends only on solid saturation S_S. As an estimate of the k_{rS}, we then take the average of the two,

$$k_{rS} = \frac{1}{2}\left[k_{rA}\left(S_A = 1 - S_S\right) + k_{rG}\left(S_G = 1 - S_S\right)\right] \tag{2.53}$$

Equation (2.50) provides a simple estimate of the permeability ϕ-factor. Then the phase effective permeabilities are computed using Equation (2.49), in which:

- $k_{r\beta}$ is computed based on the scaled saturations from Equation (2.48),

- $F_{\phi S}$ is computed from Equation (2.50),
- $k_{r\phi}$ is computed from Equation (2.51), and
- k_{rS} is computed from Equation (2.53).

The capillary pressure in the EPM #1 model is estimated using Equation (2.44), in which the various terms are computed as follows:

- ϕ_a/ϕ_{00}, computed from Equation (2.46), is used instead of ϕ/ϕ_{00},
- $k_{00}/k = 1/k_{rr} F_{\phi S}$ is computed from Equations (2.49) and (2.50),
- $k_{r\phi}$ is computed from Equation (2.51), and
- k_{rS} is computed from Equation (2.53).

Thus, the final expression for estimating the capillary pressure in the EPM #1 model is:

$$P_{cap} = \sqrt{\frac{\phi_{rr}}{k_{rr}} \cdot \frac{F_{PT}\left(1-S_S\right)}{k_{r\phi} k_{rS}}} \cdot P_{cap,00}(S^*) \tag{2.54}$$

EPM Model #2. The only difference between this model and the EPM #1 model is in the equation used to estimate the k_{rS} term, with all other equations applying unchanged. In the EPM#2 model, the quantity $F_{\phi S} = k_{r\phi}\ k_{rS}$ in Equation (2.50) is provided by Equation (2.42), leading to

$$k_{rS} = \left[\frac{\phi_0\ \left(1\text{-}S_S\right) - \phi_c}{\phi_0 - \phi_c}\right]^n \tag{2.55}$$

The term $k_{r\phi}$ is obtained from Equation (2.51). Thus, the fluid effective permeabilities in the EPM #2 model are computed from Equation (2.49). Similarly, the capillary pressure in the EPM #2 model is estimated using Equation (2.54), using k_{rS}, from Equation (2.54).

2.12.3. Pore compressibility of unconsolidated media in the presence of cementing solid phases

While the pore compressibility α_P in Equation (2.47) can be considered as a constant or even as a function of pressure during fluid flow through consolidated (lithified) porous media and/or in unconsolidated media, this approach is inadequate when cementing solid phases (such as ice and/or hydrates) are present in the pores. This is because the presence of these solid phases imparts stiffness and increases the geomechanical strength of the solid phase-impregnated porous medium, the porosity ϕ of which reacts much slower to variations in pressure P. Thus, an accurate representation of the evolution of ϕ as a function of P in these cases needs to account for the effect of the saturation S_S of such solid phases.

The most appropriate method for accounting for the effect of S_S on the porosity of unconsolidated media is by solving the coupled flow-geomechanical problem, estimating variations in P, T and phase saturation, and computing the corresponding changes in stresses and strains. These are then used to compute changes in ϕ and k. Such coupling is a possibility in the TOUGH+ code, which allows the use of the commercial geomechanical model **`FLAC3D`** (ITASCA, 2002) to evaluate the interaction between flow and geomechanical properties. This model is automatically invoked if the corresponding executable file **`FLAC3D.exe`** is present in the TOUGH+ directory and appropriate inputs are provided to the TOUGH+ code (see detailed discussion in Section 5).

If the **`FLAC3D`** model is not available or is not invoked (a frequent choice, given the large execution times required for such fully coupled flow-geomechanical problems), it is possibly to describe the effect of cementing solid phases S_S on the porosity ϕ and the

intrinsic permeability k of unconsolidated media by employing an empirical model that describes the media compressibility as:

$$\alpha_P = \exp\left\{\ln\alpha_{PL} + (\ln\alpha_{PU} - \ln\alpha_{PL})\left[1 - B_x(2.25, 2.25, S_S^*)\right]\right\}, \tag{2.56}$$

where

$$S_S^* = \frac{S_S - S_{S\min} + \delta}{S_{S\max} - S_{S\min} + 2\delta}, \tag{2.57}$$

α_{PL} is the lower limit of the medium compressibility (corresponding to the full stiffening/strengthening effect of the presence of cementing solid phases such as ice and/or hydrates), α_{PU} is the upper limit of the medium compressibility (corresponding to the absence of cementing solid phases), B_x is the incomplete beta function, S_{Smin} is the largest S_S saturation at which $\alpha_P = \alpha_{PU}$, S_{Smax} is the lowest S_S saturation at which $\alpha_P = \alpha_{PL}$, and δ is a smoothing factor. Equation (2.56) is based on geomechanical and geophysical data derived from laboratory and field observations, and results in the curve of **Figure 2.10** that scans between the α_{PU} and the α_{PL} compressibility limits.

The relative porosity ϕ/ϕ_0 is estimated from Equation (2.47), which applies unchanged, but with the composite compressibility α_P computed from Equation (2.56). **Figure 2.11** shows the relationship between the relative porosity ϕ/ϕ_0 and the pressure drop ΔP in an unconsolidated medium, and describes the cementing effect of solid phases on the medium behavior.

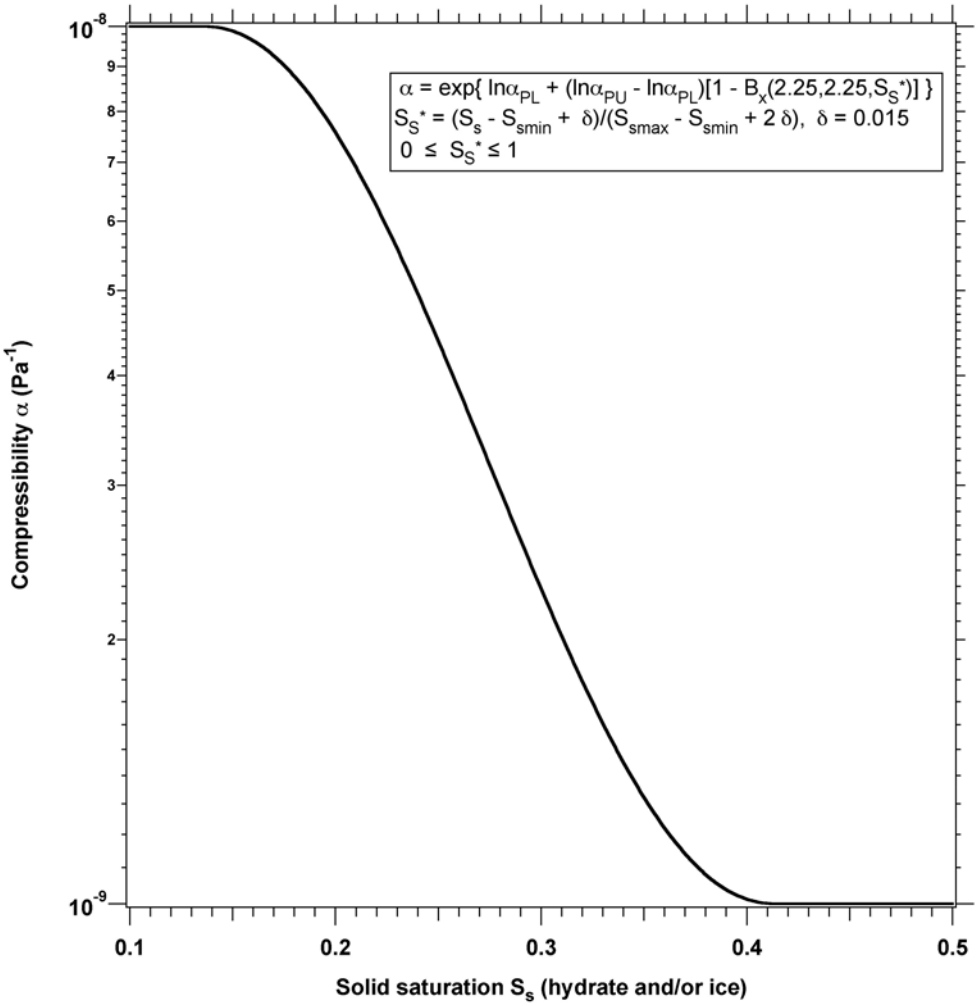

Figure 2.10. Compressibility of an unconsolidated porous medium impregnated with cementing solid phases (ice and/or hydrates). In this example, S_{Smin} = 0.15, S_{Smax} = 0.4, α_{PU} = 10^{-8} Pa^{-1}, α_{PL} = 10^{-9} Pa^{-1} and δ = 0.015.

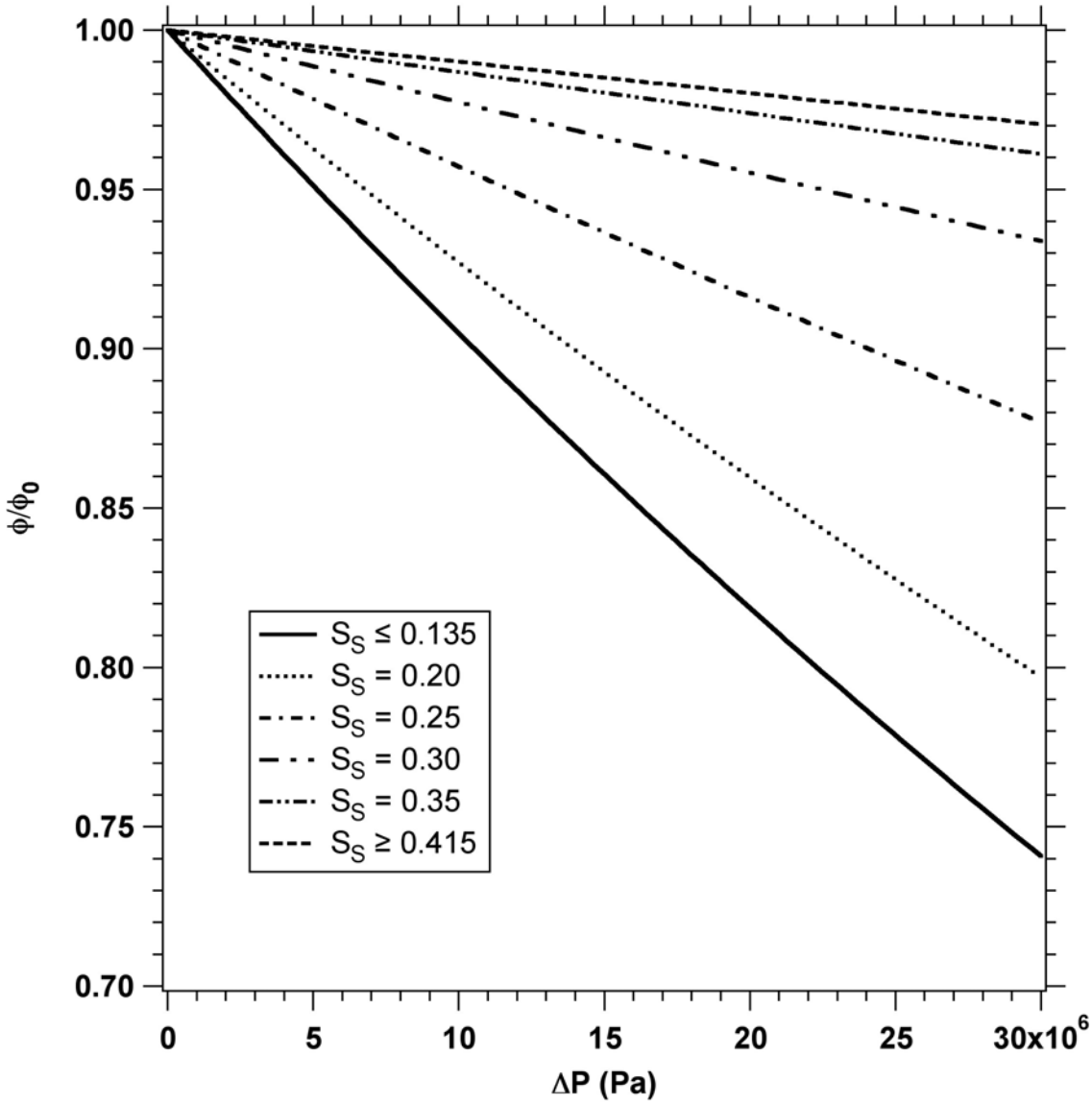

Figure 2.11. Effect of the varying compressibility described in **Figure 2.10** on the porosity of an unconsolidated porous medium undergoing depressurization for various levels of saturation S_S of cementing solid phases.

2.13. Multiphase Diffusion

2.13.1. General Considerations

Molecular diffusion plays a minor role in many subsurface flow processes, but may become a significant and even dominant mechanism for mass transport when advective velocities are small. Diffusive flux is usually written as being proportional to the gradient in the concentration of the diffusing component (Fick's law)

$$\mathbf{J} = -D\nabla C \tag{2.58}$$

where D is an effective diffusivity, which in general will depend on properties of the diffusing component, the pore fluid, and the porous medium. The concentration variable C may be chosen in a number of different ways (mass per unit volume, moles per unit volume, mass or mol fraction, etc.) [*Bird et al*., 1960; *de Marsily*, 1986].

The basic Fick's law in Equation (2.46) works well for diffusion of tracer solutes that are present at low concentrations in a single phase aqueous solution at rest with respect to the porous medium. However, many subtleties and complications arise when multiple components diffuse in a multiphase flow system. Effective diffusivities in general may depend on all concentration variables, leading to non-linear behavior especially when some components are present in significant (non-tracer) concentrations. Additional nonlinear effects arise from the dependence of tortuosity on phase saturations, and from coupling between advective and diffusive transport. For gases the Fickian model has serious limitations even at low concentrations, which prompted the development of the "dusty gas" model that entails a strong coupling between advective and diffusive transport [*Mason and Malinauskas*, 1983; *Webb*, 1998], and accounts for molecular streaming effects (Knudsen diffusion) that become very important when the mean free

path of gas molecules is comparable to pore sizes. Further complications arise for components that are both soluble and volatile, in which case diffusion in aqueous and gaseous phases may be strongly coupled via phase partitioning effects. An extreme case is the well-known enhancement of vapor diffusion in partially saturated media, which is attributed to pore-level phase change effects [*Cass et al*., 1984; *Webb and Ho*, 1998a, b].

2.13.2. Diffusion Formulation in TOUGH+

The diffusion formulation and treatment in TOUGH+ follows closely that in TOUGH2 [*Pruess et al*., 1999]. Because of the difficulties mentioned above, it is not possible to formulate a model for multiphase diffusion that would be accurate under all circumstances. We have used a pragmatic approach in which diffusive flux of component κ in phase β (= liquid, gas) is written as

$$\mathbf{J}_\beta^\kappa = -\phi\,\tau_0\,\tau_\beta\,\rho_\beta\,D_\beta^\kappa \nabla X_\beta^\kappa \tag{2.59}$$

where ϕ is the porosity, $\tau_0\tau_\beta$ is the tortuosity, which includes a porous medium dependent factor τ_0 and a coefficient that depends on phase saturation S_β, $\tau_\beta = \tau_\beta(S_\beta)$, ρ_β is the density of phase β, D_β^κ is the diffusion coefficient of component κ in bulk fluid phase β, and X_β^κ is the mass fraction of component κ in phase β. For later developments it will be useful to define a single diffusion strength factor that combines all material constants and tortuosity factors into a single effective multiphase diffusion coefficient, as follows:

$$\tilde{D}_\beta^\kappa = \phi\,\tau_0\,\tau_\beta\,\rho_\beta\,D_\beta^\kappa \tag{2.60}$$

For general two-phase conditions, the total diffusive flux is then given by

$$\mathbf{J}^\kappa = -\tilde{D}_A^\kappa \nabla X_A^\kappa - \tilde{D}_G^\kappa \nabla X_G^\kappa \tag{2.61}$$

The saturation dependence of tortuosity is not well known at present. For soils the *Millington and Quirk* [1961] model has frequently been used [*Jury et al*., 1983; *Falta et al*., 1989]

$$\tau_0 \, \tau_\beta \quad = \quad \phi^{1/3} \, S_\beta^{\;10/3} \tag{2.62}$$

which yields non-zero tortuosity coefficients as long as phase saturation is non-zero. It stands to reason that diffusive flux should vanish when a phase becomes discontinuous at low saturations, suggesting that saturation-dependent tortuosity should be related to relative permeability; e.g. $\tau_\beta(S_\beta) \approx k_{r\beta}(S_\beta)$. However, for components that partition between liquid and gas phases more complex behavior may be expected. For example, consider the case of a volatile and water-soluble compound diffusing under conditions of low gas saturation where the gas phase is discontinuous. In this case we have $k_{rG}(S_G) = 0$ (because $S_G < S_{rg}$), and $k_{rA}(S_A = 1 - S_G) < 1$, so that a model equating saturation-dependent tortuosity to relative permeability would predict weaker diffusion than in single-phase liquid conditions. For compounds with significant volatility this would be unrealistic, as diffusion through isolated gas pockets would tend to enhance overall diffusion relative to single-phase liquid conditions.

2.14. Description of Flow in Fractured Media

Figure 2.12 illustrates the classical double-porosity concept for modeling flow in fractured-porous media as developed by *Warren and Root* [1963]. Matrix blocks of low permeability are embedded in a network of interconnected fractures. Global flow in the reservoir occurs only through the fracture system, which is described as an effective porous continuum. Rock matrix and fractures may exchange fluid (or heat) locally by

means of '*interporosity flow*', which is driven by the difference in pressures (or temperatures) between matrix and fractures. Warren and Root approximated the interporosity flow as being quasi-steady, with rate of matrix-fracture interflow proportional to the difference in (local) average pressures.

The quasi-steady approximation is applicable to isothermal single-phase flow of fluids with small compressibility, where pressure diffusivities are large, so that pressure changes in the fractures penetrate quickly all the way into the matrix blocks. However, for multiphase flows, or coupled fluid and heat flows, the transient periods for interporosity flow can be very long (tens of years). In order to accurately describe such flows it is necessary to resolve the driving pressure, temperature, and mass fraction gradients at the matrix/fracture interface. In the method of "multiple interacting continua" (MINC) [*Pruess and Narasimhan*, 1982; 1985), resolution of these gradients is achieved by appropriate subgridding of the matrix blocks, as shown in **Figure 2.13**. The MINC concept is based on the notion that changes in fluid pressures, temperatures, phase compositions, etc., due to the presence of sinks and sources (production and injection wells) will propagate rapidly through the fracture system, while invading the tight matrix blocks only slowly. Therefore, changes in matrix conditions will (locally) be controlled by the distance from the fractures. Fluid and heat flow from the fractures into the matrix blocks, or from the matrix blocks into the fractures, can then be modeled by means of one-dimensional strings of nested grid blocks, as shown in **Figure 2.13**.

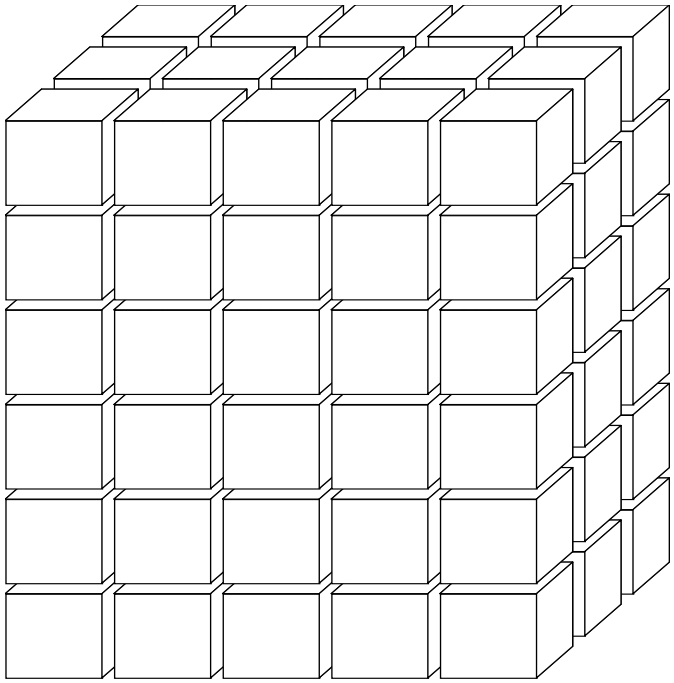

Figure 2.12. Idealized double porosity model of a fractured porous medium.

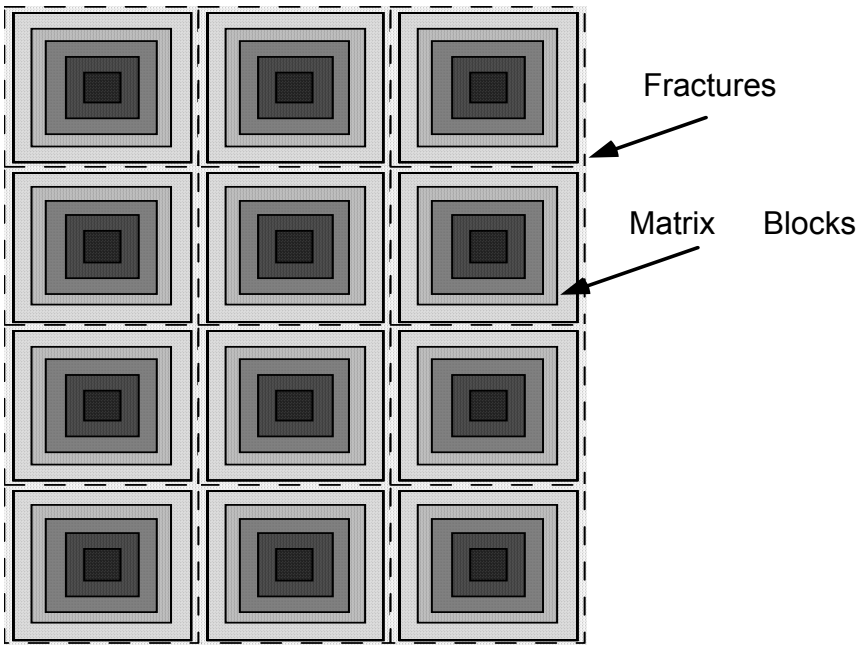

Figure 2.13. Subgridding in the method of "multiple interacting continua" (MINC).

In general it is not necessary to explicitly consider subgrids in all of the matrix blocks separately. Within a certain reservoir subdomain (corresponding to a finite difference grid block), all fractures will be lumped into continuum # 1, all matrix material within a certain distance from the fractures will be lumped into continuum # 2, matrix material at larger distance becomes continuum # 3, and so on. Quantitatively, the subgridding is specified by means of a set of volume fractions `VOL(j)`, `j = 1, ..., J`, into which the primary porous medium grid blocks are partitioned. The MINC-process in the **`MeshMaker.f95`** (a companion code distributed with TOUGH+, see Section 7) operates on the element and connection data of a porous medium mesh to calculate, for given data on volume fractions, the volumes, interface areas, and nodal distances for a secondary fractured medium mesh. The information on fracturing (spacing, number of sets, shape of matrix blocks) required for this is provided by a *proximity function* PROX(x) which expresses, for a given reservoir domain V_o, the total fraction of matrix material within a distance x from the fractures. If only two continua are specified (one for fractures, one for matrix), the MINC approach reduces to the conventional double-porosity method. Full details are given in a separate report [*Pruess*, 1983].

The MINC-method as implemented in the **`MeshMaker.f95`** code can also describe global matrix-matrix flow. **Figure 2.14** shows the most general approach, often referred to as *dual permeability*, in which global flow occurs in both fracture and matrix continua. It is also possible to permit matrix-matrix flow only in the vertical direction. For any given fractured reservoir flow problem, selection of the most appropriate gridding scheme must be based on a careful consideration of the physical and geometric

conditions of flow. The MINC approach is not applicable to systems in which fracturing is so sparse that the fractures cannot be approximated as a continuum.

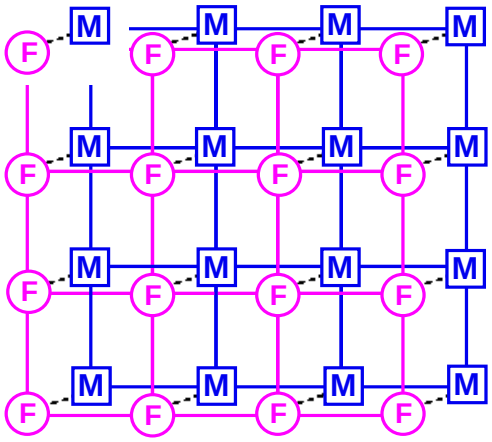

Figure 2.14. Flow connections in the “dual permeability” model. Global flow occurs between both fracture (F) and matrix (M) grid blocks. In addition there is F-M interporosity flow.

3.0. Design and Implementation of TOUGH+HYDRATE

3.1. Primary Variables

The thermodynamic state and the distribution of the mass components among the four possible phases are determined from the hydrate equation of state. Following the standard approach employed in the TOUGH2 [*Pruess et al.*, 1999] family of codes, in **T+H** the system is defined uniquely by a set of N_κ primary variables (where κ denotes the number of mass and heat components under consideration, see Section 2.2) that completely specifies the thermodynamic state of the system. Although the number N_k of the primary variables is initially set at the maximum expected in the course of the simulation and does not change during the simulation, the thermodynamic quantities used

as primary variables can change in the process of simulation to allow for the seamless consideration of emerging or disappearing phases and components.

A total of 26 states (phase combinations) covering the entire phase diagram in Figure 2.5 are described in **T+H**. Of those, 13 correspond to the equilibrium hydration reaction option, and 13 to the kinetic hydration reaction option. The primary variables used for the various phase states without inhibitor are listed in **Tables 3.1** and **3.2**, respectively. For systems with an inhibitor, the additional primary variable is *X_i_A*, (corresponding to X_A^i, i.e., the mass fraction of the inhibitor in the aqueous phase). The option set for either equilibrium or kinetic hydration reactions is complete, although some of the phase states are only possible under laboratory conditions and difficult to reach under any conditions of gas production from dissociating natural hydrate deposits.

The primary variables in **Tables 3.1** and **3.2** are necessary and sufficient to uniquely define the H_2O-CH_4 system. Note that the lack of equilibrium in the kinetic model introduces an additional degree of freedom, and consequently necessitates an additional equation. This requires special care to ensure that the resulting system is not over-defined when the hydrate and/or gas phases are not present.

Table 3.1. Primary Variables in Equilibrium Hydrate Simulations without Inhibitor*.

Phase	State Identifier	Primary Variable 1	Primary Variable 2	Primary Variable 3
1-Phase: G	**Gas**	*P_gas*	*Y_m_G*	*T*
1-Phase: A	**Aqu**	*P*	*X_m_A*	*T*
2-Phase: A+G	**AqG**	*P_gas*	*S_aqu*	*T*
2-Phase I+G	**IcG**	*P_gas*	*S_ice*	*T*
2-Phase H+G	**GsH**	*P_gas*	*S_gas*	*T*
2-Phase: A+H	**AqH**	*P*	*S_aqu*	*T*
2-Phase: A+I	**AqI**	*P*	*S_aqu*	*X_m_A*
2-Phase: I+H	**IcH**	*P*	*S_ice*	*T*
3-Phase: A+H+G	**AGH**	*S_gas*	*S_aqu*	*T*
3-Phase: A+I+G	**AIG**	*P_gas*	*S_aqu*	*S_gas*
3-Phase: A+I+H	**AIH**	*P*	*S_aqu*	*S_ice*
3-Phase: I+H+G	**IGH**	*S_gas*	*S_ice*	*T*
Quadruple Point: I+H+A+G	**QuP**	*S_gas*	*S_aqu*	*S_ice*

Where the possible primary variables are: *P*, pressure [Pa]; *P_gas*, gas pressure [Pa]; *T*, temperature [C]; *X_m_A*, mass fraction of CH_4 dissolved in the aqueous phase [-]; *Y_m_G*, mass fraction of CH_4 dissolved in the gas phase [-]; *S_aqu*, liquid saturation [-]; *S_gas*, gas saturation [-]; *S_ice*, ice saturation [-]; *X_i_A*, mass fraction of inhibitor dissolved in the aqueous phase [-].

*For inhibitor: *X_i_A* becomes the 3rd primary variable, and the 3rd primary variable (as listed in this table) becomes the 4th primary variable.

Table 3.2. Primary Variables in Kinetic Hydrate Simulations without Inhibitor*.

Phase	State Identifier	Primary Variable 1	Primary Variable 2	Primary Variable 3	Primary Variable 4
1-Phase: G	**Gas**	*P_gas*	*Y_m_G*	*S_hyd*	*T*
1-Phase: A	**Aqu**	*P*	*X_m_A*	*S_hyd*	*T*
2-Phase: A+G	**AqG**	*P_gas*	*S_aqu*	*S_hyd*	*T*
2-Phase I+G	**IcG**	*P_gas*	*S_ice*	*S_hyd*	*T*
2-Phase H+G	**GsH**	*P_gas*	*S_gas*	*S_ice*	*T*
2-Phase: A+H	**AqH**	*P*	*S_aqu*	*X_m_A*	*T*
2-Phase: A+I	**AqI**	*P*	*S_aqu*	*X_m_A*	*T*
2-Phase: I+H	**IcH**	*P*	*S_ice*	*S_gas*	*T*
3-Phase: A+H+G	**AGH**	*P_gas*	*S_aqu*	*S_gas*	*T*
3-Phase: A+I+G	**AIG**	*P_gas*	*S_aqu*	*S_hyd*	*S_gas*
3-Phase: A+I+H	**AIH**	*P*	*S_aqu*	*S_ice*	*T*
3-Phase: I+H+G	**IGH**	*P_gas*	*S_gas*	*S_ice*	*T*
Quadruple Point: I+H+A+G	**QuP**	*P_gas*	*S_aqu*	*S_gas*	*S_ice*

Where the possible primary variables are: *P*, pressure [Pa]; *P_gas*, gas pressure [Pa]; *T*, temperature [C]; *X_m_A*, mass fraction of CH_4 dissolved in the aqueous phase [-]; *Y_m_G*, mass fraction of CH_4 dissolved in the gas phase [-]; *S_aqu*, liquid saturation [-]; *S_gas*, gas saturation [-]; *S_hyd*, hydrate saturation [-]; *S_ice*, ice saturation [-]; *X_i_A*, mass fraction of inhibitor dissolved in the aqueous phase [-].

*For inhibitor: *X_I_A* becomes the 4th primary variable, and the 4th primary variable (as listed in this table) becomes the 5th primary variable.

3.2. Space and Time Discretization

The continuum equations (2.3) are discretized in space using the integral finite difference method (IFD) [*Edwards*, 1972; *Narasimhan and Witherspoon*, 1976]. Introducing appropriate volume averages, we have

$$\int_{V_n} M \, dV \quad = \quad V_n \, M_n \tag{3.1}$$

where M is a volume-normalized extensive quantity, and M_n is the average value of M over V_n. Surface integrals are approximated as a discrete sum of averages over surface segments A_{nm}:

$$\int_{\Gamma_n} \mathbf{F}^{\kappa} \bullet \mathbf{n} \, d\Gamma \quad = \quad \sum_m A_{nm} \, F_{nm} \tag{3.2}$$

Here F_{nm} is the average value of the (inward) normal component of **F** over the surface segment A_{nm} between volume elements V_n and V_m. The discretization approach used in the integral finite difference method and the definition of the geometric parameters are illustrated in **Figure 3.1**.

The discretized flux is expressed in terms of averages over parameters for elements V_n and V_m. For the basic Darcy flux term in Equation (2.15), we have

$$F_{\beta,nm} \quad = \quad -k_{nm} \left[\frac{k_{r\beta} \, \rho_\beta}{\mu_\beta} \right]_{nm} \left[\frac{P_{\beta,n} - P_{\beta,m}}{D_{nm}} - \rho_{\beta,nm} \, g_{nm} \right] \tag{3.3}$$

where the subscripts (nm) denote a suitable averaging at the interface between grid blocks n and m (interpolation, harmonic weighting, upstream weighting). $D_{nm} = D_n + D_m$ is the distance between the nodal points n and m, and g_{nm} is the component of gravitational

acceleration in the direction from m to n. Discretization of diffusive fluxes raises some subtle issues, and is discussed separately in Section 3.5.

Substituting Equations. (3.l) and (3.2) into the governing Equation (2.3), a set of first-order ordinary differential equations in time is obtained.

$$\frac{d M_n^\kappa}{d t} = \frac{1}{V_n} \sum_m A_{nm} F_{nm}^\kappa + q_n^\kappa \tag{3.4}$$

Time is discretized as a first-order finite difference, and the flux and sink and source terms on the right-hand side of Equation (3.4) are evaluated at the new time level, $t^{k+1} = t^k + \Delta t$, to obtain the numerical stability needed for an efficient calculation of strongly nonlinear problems (such as the ones involving multiphase flow and phase changes). This treatment of flux terms is known as *fully implicit*, because the fluxes are expressed in terms of the unknown thermodynamic parameters at time level t^{k+1}, so that these unknowns are only implicitly defined in the resulting equations [*Peaceman*, 1977].

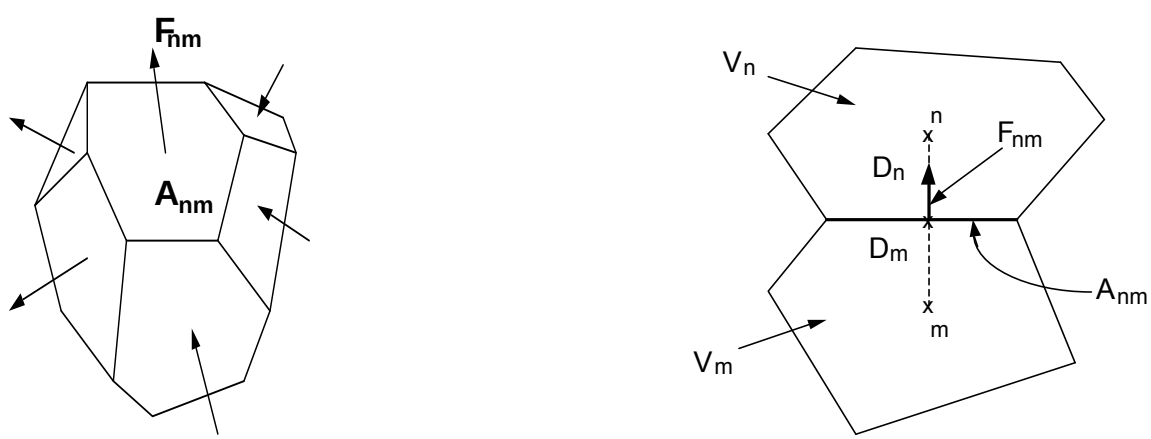

Figure 3.1. Space discretization and geometry data in the integral finite difference method.

The time discretization results in the following set of coupled non-linear, algebraic equations

$$R_n^{\kappa,k+1} = M_n^{\kappa,k+1} \; - \; M_n^{\kappa,k} - \frac{\Delta t}{V_n} \left(\sum_m A_{nm} F_{nm}^{\kappa,k+1} \; + \; V_n q_n^{\kappa,k+1} \right) \; = \; 0 \qquad (3.5)$$

where we have introduced residuals $R_n^{\kappa,k+1}$. For each volume element (grid block) V_n, there are N_κ equations, so that for a system discretized into N_E grid blocks, Equation (3.5) represents a total of $N_\kappa \times N_E$ coupled non-linear equations.

3.3. The Newton-Raphson Iteration

The unknowns of Equation (3.5) are the $N_\kappa \times N_E$ independent primary variables $\{x_i;$ i = 1, ..., $N_\kappa \times N_E\}$ which completely define the state of the flow system at time level t^{k+l}. These equations are solved by Newton/Raphson iteration, which is implemented as follows. We introduce an iteration index p and expand the residuals $R_n^{\kappa,k+1}$ in Equation (3.5) at iteration step $(p + 1)$ in a Taylor series in terms of those at index p, i.e.,

$$R_n^{\kappa,k+1}\left(x_{i,p+1}\right) = R_n^{\kappa,k+1}\left(x_{i,p}\right) \; + \; \sum_{\mathrm{i}} \left. \frac{\partial R_n^{\kappa,k+1}}{\partial x_i} \right|_p \left(x_{i,p+1} - x_{i,p}\right) \; + \; ... \; = \; 0 \qquad (3.6)$$

Retaining only terms up to first order, we obtain a set of $N_\kappa \times N_E$ linear equations for the increments $\left(x_{i,p+1} - x_{i,p}\right)$:

$$- \sum_{\mathrm{i}} \left. \frac{\partial R_n^{\kappa,k+1}}{\partial x_i} \right|_p \left(x_{i,p+1} - x_{i,p}\right) \; + \; ... \; = \; R_n^{\kappa,k+1}\left(x_{i,p}\right) \qquad (3.7)$$

All terms $\partial R_n/\partial x_i$ in the Jacobian matrix are evaluated by numerical differentiation.

Equation (3.7) is solved by sparse direct matrix methods or iteratively by means of preconditioned conjugate gradients [*Moridis and Pruess*, 1995; 1998; Pruess et al., 1999]. Iteration is continued until the residuals $R_n^{\kappa,k+1}$ are reduced below a preset convergence tolerance according to:

$$\left| \frac{R_{n,p+1}^{\kappa,k+1}}{M_{n,p+1}^{\kappa,k+1}} \right| \leq \varepsilon_1 \tag{3.8}$$

The default (relative) convergence criterion is $\varepsilon_1 = 10^{-5}$ (TOUGH+ input parameter RE1, see Section 10). When the accumulation terms are smaller than ε_2 (TOUGH+ input parameter RE2, default $\varepsilon_2 = 1$, see Section 10), an absolute convergence criterion is imposed,

$$\left| R_{n,p+1}^{\kappa,k+1} \right| \leq \varepsilon_1 \varepsilon_2 \tag{3.9}$$

The number of iterations to convergence varies with the nonlinearity of the problem. For well-behaved problems, convergence is usually attained in 3--4 iterations. If convergence cannot be achieved within a certain number of iterations (default 8, see Section 10), the time step size Δt is reduced and a new iteration process is started.

3.4. Implications of the Space Discretization Approach

It is appropriate to add some comments about the space discretization technique in TOUGH+. The entire geometric information of the space discretization in Equation (3.5) is provided in the form of a list of grid block volumes V_n, interface areas A_{nm}, nodal distances D_{nm} and components g_{nm} of gravitational acceleration along nodal lines. There

is no reference whatsoever to a global system of coordinates, or to the dimensionality of a particular flow problem.

The discretized equations are in fact valid for arbitrary irregular discretizations in one, two or three dimensions, and for porous as well as for fractured media. This flexibility should be used with caution, however, because the accuracy of solutions depends upon the accuracy with which the various interface parameters in equations such as (3.3) can be expressed in terms of average conditions in grid blocks. A general requirement is that there exists approximate thermodynamic equilibrium in (almost) all grid blocks at (almost) all times [*Pruess and Narasimhan*, 1985]. For systems of regular grid blocks referenced to global coordinates in cylindrical (*r,z*) and/or Cartesian (*x,y,z*) systems, Equation (3.5) is identical to a conventional finite difference formulation [*Peaceman*, 1977; *Moridis and Pruess*, 1992].

3.5. Space Discretization of Diffusive Fluxes

Space discretization of diffusive flux in multiphase conditions raises some subtle issues. A finite difference formulation for total diffusive flux, Equation (3.10), may be written as

$$\left(\mathbf{J}^{\kappa}\right)_{nm} = -\left(\tilde{D}_A^{\kappa}\right)_{nm} \frac{\left(X_A^{\kappa}\right)_m - \left(X_A^{\kappa}\right)_n}{D_{nm}} - \left(\tilde{D}_G^{\kappa}\right)_{nm} \frac{\left(X_G^{\kappa}\right)_m - \left(X_G^{\kappa}\right)_n}{D_{nm}} \tag{3.10}$$

This expression involves the as yet unknown diffusive strength coefficients $(\Sigma_A^{\kappa})_{nm}$ and $(\Sigma_G^{\kappa})_{nm}$ at the interface, which must be expressed in terms of the strength coefficients in the participating grid blocks. Invoking conservation of diffusive flux across the interface between two grid blocks leads in the usual way to the requirement of harmonic weighting

of the diffusive strength coefficients. However, such weighting may in general not be applied separately to the diffusive fluxes in gas and liquid phases, because these may be strongly coupled by phase partitioning effects. This can be seen by considering the extreme case of diffusion of a water-soluble and volatile compound from a grid block in single-phase gas conditions to an adjacent grid block that is in single-phase liquid conditions. Harmonic weighting applied separately to liquid and gas diffusive fluxes would result in either of them being zero, because for each phase effective diffusivity is zero on one side of the interface. Thus total diffusive flux would vanish in this case, which is unphysical. In reality, tracer would diffuse through the gas phase to the gas-liquid interface, would establish a certain mass fraction in the aqueous phase by dissolution, and would then proceed to diffuse away from the interface through the aqueous phase. Similar arguments can be made in the less extreme situation where liquid saturation changes from a large to a small value rather than from 1 to 0, as may be the case in the capillary fringe, during infiltration events, or at fracture-matrix interfaces in variably saturated media.

TOUGH+ features the fully coupled approach employed in TOUGH2 [*Pruess et al.*, 1999], in which the space-discretized version of Equation (3.10) of the total multiphase diffusive flux Equation (2.49) is re-written in terms of an effective multiphase diffusive strength coefficient and a single mass fraction gradient. Choosing the liquid mass fraction for this we have

$$\left(\mathbf{f}^{\kappa}\right)_{nm} = -\left[\left(\sum\nolimits_{A}^{\kappa}\right)_{nm} + \left(\sum\nolimits_{G}^{\kappa}\right)_{nm} \frac{\left(X_G^{\kappa}\right)_m - \left(X_G^{\kappa}\right)_n}{\left(X_A^{\kappa}\right)_m - \left(X_A^{\kappa}\right)_n}\right] \frac{\left(X_A^{\kappa}\right)_m - \left(X_A^{\kappa}\right)_n}{D_{nm}}, \qquad (3.11)$$

where the gas phase mass fraction gradient has been absorbed into the effective diffusive strength term (in braces). As is well known, flux conservation at the interface then leads to the requirement of harmonic weighting for the full effective strength coefficient. In order to be able to apply this scheme to the general case where not both phases may be present on both sides of the interface, we always define both liquid and gas phase mass fractions in all grid blocks, regardless of whether both phases are present. Mass fractions are assigned in such a way as to be consistent with what would be present in an evolving second phase. This procedure is applicable to all possible phase combinations, including the extreme case where conditions at the interface change from single phase gas to single phase liquid. Note that, if the diffusing tracer exists in just one of the two phases, harmonic weighting of the strength coefficient in Equation (3.11) will reduce to harmonic weighting of either Σ_A^κ or Σ_G^κ.

The simpler scheme of separate harmonic weighting for individual phase diffusive fluxes is retained as an option.

3.6. Compiling the TOUGH+HYDRATE Code

TOUGH+HYDRATE is written in standard FORTRAN 95/2003. It has been designed for maximum portability, and runs on any computational form (Unix and Linux workstations, PC, Macintosh) for which such compilers are available. Running **T+H** involves compilation and linking of the following code segments and in the following order:

(1) T_Hydrate_Definitions.f95 (*)

Segment providing default parameter values describing the basic attributes of the equation of state (i.e., number of components, number of phases, etc.)

(2) **T_Allocate_Memory.f95**

Segment responsible for the dynamic memory allocation (following input describing the size of the problem) and dimensioning of most arrays needed by the code, in addition to memory deallocation of unnecessary arrays.

(3) **T_Untility_Functions.f95**

Segment that includes all utility functions (including a wide variety of mathematical functions, table interpolation routines, sorting algorithms, etc.).

(4) **T_H2O_Properties.f95**

Segment that includes (a) all the water-related constants (parameters), and (b) procedures describing the water behavior and thermophysical properties/processes in its entire thermodynamic phase diagram.

(5) **T_Media_Properties.f95**

Segment that describes the hydraulic and thermal behavior of the geologic medium (porous or fractured), i.e., capillary pressure and relative permeability under multiphase conditions, interface permeability and mobility, and interface thermal conductivity.

(6) **T_RealGas_EOS.f95**

Segment that includes (a) the important constants (parameters) of all hydrate-forming gases, and (b) procedures describing the equation of state (EOS) of real gases (pure or mixtures) using any of the Peng-Robinson, Redlich-Kwong, or Soave-Redlich-Kwong cubic EOS model. The

procedures in this segment compute the following parameters and processes: compressibility, density, fugacity, enthalpy (ideal and departure), internal energy (ideal and departure), entropy (ideal and departure), thermal conductivity, viscosity, binary diffusion coefficients, solubility in water, and heat of dissolution in water.

(7) T_Hydrate_Properties.f95 (*)

Segment that describes the properties and processes of the CH_4-hydrate. It includes procedures that describe the *P-T* relationship along the three-phase equilibrium regimes of the CH_4-hydrate phase diagram (see Figure 2.4) and compute the heat of the hydration reaction, the hydrate density and enthalpy, and the reaction rate when kinetic formation/dissociation are invoked.

(8) T_Geomechanics.f95 (*)

Segment that describes the geomechanically-induced changes on the flow properties of the porous media. These include porosity ϕ changes caused by pressure and/or temperature variations, intrinsic permeability k changes caused by porosity changes, and scaling of capillary pressures P_{cap} to reflect changes in ϕ and k. The ϕ and k changes are computed using either simplified of full geomechanical models. When the simplified model is invoked, ϕ is a function of (a) P and the pore compressibility α_P and (b) of T and the pore thermal expansivity α_T, while (c) k changes are estimated using emprirical relationships (see Section 8). Changes in ϕ and k can also be computed by using a full geomechanical model, which can be optionally coupled with TOUGH+.

(9) T_Hydrate_Specifics.f95 (*)

Segment that includes procedures specific to the hydrate simulation, such as the reading of hydrate-specific inputs, the preparation of hydrate-specific output files, and the computation of the thermal conductivity in

hydrate-bearing media. Generic procedures and operator extension -- which override (*overload*) the standard procedures used by TOUGH+ for the simulation of non-hydrate problems – are defined in this segment, which does not include any procedures describing the hydrate equation of state.

(10) T_Main.f95

Main program that organizes the calling sequence of the high-level events in the simulation process, and includes the writing of important general comments in the standard output files, timing procedures, and handling of files needed by the code and/or created during the code execution.

(11) T_Hydrate_EOS.f95 (*)

Segment that describes the equation of state of the CH_4-hydrate, assigns initial conditions, computes the thermophysical properties of the hydrate-bearing medium, and determines phase changes and the state of the system from the 22 possible options (see Section 3.1). This segment also includes the procedure that computes the elements of the Jacobian matrix for the Netwon-Raphson iteration (see Sections 3.3).

(12) T_Matrix_Solvers.f95

A linear algebra package that includes all the direct and iterative solvers availabse in TOUGH+ (see Section 10).

(13) T_Executive.f95

The executive segment of TOUGH+. It includes the procedures that advance the time in the simulation process, estimate the time-step size for optimum performance, populate the matrix arrays and invoke the solvers of the Jacobian, invoke special linear algebra for matrix pre-processing in cases of very demanding linear algebra problems, compute mass and energy balances, compute rates in sources and sinks, compute binary

diffusion coefficients, write special output files, and conduct other miscellaneous operations.

(14) `T_Inputs.f95`

This segment includes the procedures involved in the reading of the general input files needed for TOUGH+ simulations. It does not include any procedure reading hydrate-related data (this is accomplished in the **`TFx_Hydrate_Spcfc.f95`** segment).

The segments denoted by (*) are specific to the hydrate problem, and are needed by **T+H**. All other segments are common to all TOUGH+ simulations.

Additionally, **T+H** is distributed with the **`Meshmaker.f95`** FORTRAN code, which used to be part of the main code in the TOUGH and TOUGH2 simulators, but is a separate entity in the TOUGH+ family of codes. **`Meshmaker.f95`** is used for the space discretization (gridding) of the domain of the problem under study (see Section 7).

NOTE: In compiling **T+H** v1.2, *it is important that the free-format source code option be invoked for proper compilation*.

4.0. Input Data Requirements

In this section, we discuss in detail the input requirements for TOUGH+HYDRATE v1.2 simulations. To ensure backward compatibility with older TOUGH2 input files (a functional requirement for TOUGH+), the input data for **T+H** are a superset of the input data used in conventional TOUGH2 simulations. Thus, the inputs for the current version of TOUGH+ are similar in type, input format, and parameter representation with those in TOUGH2 (an advanced, keyword-based data input system, will be made available in future TOUGH+ releases).

4.1. Input Procedure

The input procedure in the current version of TOUGH+ remains similar in many aspects to that of TOUGH2 [*Pruess et al.*, 1991]. Input data can be provided in a flexible manner by means of one or several ASCII data files. The TOUGH+ inputs are in

standard metric (SI) units, such as meters, seconds, kilograms, ˚C, and the corresponding derived units, such as Newtons, Joules, and Pascal = N/m^2 for pressure.

In the TOUGH+ input file data are organized in data blocks that are defined by keywords. Quite often, only the first five characters of the keywords typed in columns 1-5 (see **Table 4.1**) are read, because these are sufficient to recognize the data block. While the contents of the various blocks are described in detail in Sections 5-12, here we describe some important records/keywords, and provide some general comments about their occurrence and arrangement in the input file.

4.1.1. Data Block/Keyword **TITLE**

The first record of the input file in any TOUGH+ simulation is **TITLE**, which includes a header of up to 132 characters. This record is necessary for any simulation to begin.

4.1.2. Keyword/Record **START** *or* **RANDO**MN

The use of the **START** or **RANDO**MN record/keyword typed in columns 1-5 allows a more flexible initialization (the boldface in **RANDO**MN indicates the portion of the keyword used by the code). More specifically, when **START** or **RANDO**MN is present, data initialization in the **INCON** data block (or file) can be in arbitrary order, and need not be present for all grid blocks (in which case defaults will be used). Without **START** or **RANDO**MN, there must be a one-to-one correspondence between the data in data blocks **ELEME** and **INCON**.

4.1.3. Keyword/Record **ENDCY**

A record with the **ENDCY** keyword typed in columns 1-5 closes the TOUGH+ input file and instructs the code to initiate the simulation.

4.1.4. Keyword/Record **ENDFI**

The presence of the **ENDFI** keyword in columns 1-5 is an alternative (to **ENDCY**) ending keyword in a TOUGH+ standard input file. Presence of **ENDFI** keyword causes the simulation to be skipped after printing basic input information. This is a useful option when the simulation is limited to an attempt to obtain some basic information of the properties and conditions of the system in its initial state.

4.1.5. General Comments on the Structure of TOUGH+ Standard Input Files

Every TOUGH+ input file must (a) begin with the record/keyword **TITLE** (providing a problem title of up to 80 characters) and (b) end either with the record **ENDCY**, or, alternatively, with the record **ENDFI** (if no flow simulation is to be carried out). Data records beyond **ENDCY** (or **ENDFI**) will be ignored.

Some data blocks, such as those specifying reservoir domains (**ROCKS or MEDIA**), volume elements (**ELEME**), connections (**CONNE**), and sinks/sources (**GENER**), have a variable number of records, while others have a fixed number of records. The end of variable-length data blocks is indicated by a blank record. The data blocks between the **TITLE** and **ENDCY**/**ENDFI** keywords can be provided in arbitrary order, except for the

data block **ELEME**, which must precede (if either is present) the blocks **CONNE** and **EXT-INCON**. The blocks **ELEME** and **CONNE** must either be both provided through the standard input file, or must both be absent, in which case alternative means for specifying geometry data will be employed (see Section 7). The data block **GENER** can be omitted if there are no sinks and sources in the problem. If the keyword **START** or **RANDO**MN is present (see Section 4.1.2), the data block **INCON** can be incomplete, with elements (grid blocks) in arbitrary order, or can be absent altogether.

Elements for which no initial conditions are specified in **INCON** will then be assigned domain-specific initial conditions from (a) block **INDOM** (if present), or (b) from the data block **EXT-INCON** (if present), or (c) from the 'generic' initial conditions given in block **PARAM**, along with default porosities given in block **ROCKS**. If **START** or **RANDO**MN is not present, **INCON** must contain information for all elements, in exactly the same order as they are listed in block **ELEME**.

Between data blocks, the standard TOUGH+ input file may include an arbitrary number of records that do not begin with any of the TOUGH+ keywords. This is useful for inserting comments about problem specifications directly into the input file. TOUGH+ will gather all these comments and will print the first 50 such records in the output file.

Much of the data handling in TOUGH+ is accomplished by means of disk files, which can be edited and modified using any text editor. The initialization of the arrays for geometry, generation, and initial condition data is always made from the disk files **MESH** (or **MINC**), **GENER**, and **INCON**. A user can either provide these files at execution

time, or they can be written from TOUGH+ input data during the initialization phase of the program.

If the data blocks **GENER** and/or **INCON** are not provided in the standard input file, and if no disk files **GENER** and/or **INCON** are present, defaults will take effect (no generation; initial conditions from block **INDOM**, or from block **EXT-INDOM**, or defaults from block **PARAM**). To ensure that these defaults are used, the disk files **GENER** and/or **INCON** (from a previous run) must be removed from the execution environment/directory. A safe way to use default generation and initial conditions is to specify *dummy* data blocks in the input file, consisting of one record with **GENER** or **INCON**, followed by a blank record.

The format of data blocks **ELEME**, **CONNE**, **GENER**, and **INCON** is basically the same (see Section 7) when these data are provided as disk files and when they are given as part of the input file. However, specification of these data through the input file rather than as disk files offers some added conveniences, which are useful when a new simulation problem is initiated. For example, a sequence of identical items (volume elements, connections, sinks or sources) can be specified through a single data record. Also, indices used internally for cross-referencing elements, connections, and sources will be generated internally by TOUGH+ rather than having them provided by the user. **INCON**, **GENER**, and **INCON** disk files written by TOUGH+ can be merged into an input file without changes, keeping the cross-referencing information.

Table 4.1. TOUGH+ input data blocks.

Keyword	Sec.	Function
TITLE (1st record)	**4.1.1**	Data record (single line) with simulation title
MEMORY	**5.1**	Dynamic memory allocation
HYDRATE	**6.1**	Parameters describing hydrate properties and behavior
ROCKS or **MEDIA**	**6.2**	Hydrogeologic parameters for various reservoir domains
RPCAP or **WETTA**BILITY	**6.3**	Optional; parameters for relative permeability and capillary pressure functions
DIFFUSION	**6.4**	Optional; diffusivities of mass components
***ELEME**	**7.1**	List of grid blocks (volume elements)
***CONNE**	**7.2**	List of flow connections between grid blocks
INDOM	**8.1**	Optional; list of initial conditions for specific reservoir domains
***INCON**	**8.2**	Optional; list of initial conditions for specific grid blocks
EXT-INCON	**8.3**	Optional; list of initial conditions for specific grid blocks
BOUNDARIES	**8.6**	Optional; provides time-variable conditions at specific boundaries
***GENER**	**9.1**	Optional; list of mass or heat sinks and sources
PARAM	**10.1**	Computational parameters; time stepping and convergence parameters; program options
SOLVR	**10.2**	Optional; specifies parameters used by linear equation solvers.
TIMES	**11.2**	Optional; specification of times for generating printout
ELEM_TIME_SERIES	**11.3**	Optional; specifies grid blocks for desired time series data
CONX_TIME_SERIES	**11.4**	Optional; specifies connections for desired time series data
SS_TIME_SERIES	**11.5**	Optional; specifies sinks/sources for desired time series data
SUBDOMAINS	**11.6**	Optional; specifies grid subdomains for desired time series data
NOVER	**11.6**	Optional; suppresses printout of version numbers and dates
START or **RANDO**MN	**4.1.2**	Optional; one data record for more flexible initialization
ENDCY(last record)	**4.1.3**	Record closes TOUGH+ input file and initiates simulation
ENDFI(last record)	**4.1.4**	Alternative for closing TOUGH+ input file which causes flow simulation to be skipped.

Blocks labeled with * can be provided as separate disk files and omitted from input file.

5.0. Memory Specification and Allocation

5.1. Data Block MEMORY

This is the data block that reads the data for the dynamic memory allocation in the **T+H** simulation. This block must always follow the TITLE record.

Record MEMORY.1

This record must always include the following header:

MEMORY

Record MEMORY.2

Reads the character variable EOS_Name according to FORMAT(A15). For hydrate simulations, the value of EOS_Name can be either HYDRATE-EQUILIBRIUM or HYDRATE-KINETIC. Note that this parameter does not necessarily provide an indication of the type of hydrate simulation (equilibrium vs. kinetic) to follow, but is used strictly to allocate memory to the pertinent storage arrays.

Record MEMORY.3

The following information is provided in MEMORY.3 using a free format:

```
NumCom, NumEq, NumPhases, binary_diffusion
```

These parameters are defined as follows:

NumCom	Integer denoting the number of mass components (see Section 2)
NumEq	Integer denoting the number of equations
NumPhases	Integer denoting the number of phases. Note that for in the case of hydrates, the code resets internally the input value of NPH to 4. This ensures that the simulation can fully describe the evolution of the hydrate-bearing system should ice emerge.
binary_diffusion	Logical variable indicating whether binary diffusion is to be considered (binary_diffusion = .TRUE.) or ignored (binary_diffusion = .FALSE.).

The following combinations are possible for hydrate simulations in **T+H**:

(1) (NumCom, NumEq, NumPhases) = (2,3,4):
Equilibrium hydrate reaction, no inhibitor

(2) (NumCom, NumEq, NumPhases) = (3,4,4):
Equilibrium hydrate reaction with inhibitor

(3) (NumCom, NumEq, NumPhases) = (3,4,4):
Kinetic hydrate reaction, no inhibitor

(4) (NumCom, NumEq, NumPhases) = (4,5,4):
Kinetic hydrate reaction with inhibitor

Record MEMORY.4

In MEMORY.4, the following parameters are read using a free format:

```
coordinate_system, Max_NumElem,
Max_NumConx, ElemNameLength,
active_conx_only, boundaries_in_matrix
```

These parameters are defined as follows:

`coordinate_system` Character variable describing the coordinate system used in the study. It can assume the values `'Cartesian'` or `'Cylindrical'`.

`Max_NumElem` Integer variable defining the maximum number of elements (cells, gridblocks) in the discretized simulation domain

`Max_NumConx` Integer variable defining the maximum number of connections in the discretized simulation domain.

Information on the concepts of elements and connections can be found in Section 8, and in *Pruess et al.* [1999]. As a general rule, `MaxNum_Conx > ND*MaxNum_Elem`, where ND is the dimensionality of the problem.

`ElemNameLength` Integer variable defining the number of characters in the element names. It may be either 5 or 8 (default is 5 in v1.2).

`active_conx_only` Logical variable indicating whether the simulation will be halted after determining the active connections in the grid. This feature is useful when running a simulation that uses a subset of the elements of a large grid without correspondingly adjusting the connections, and is designed to reduce the very large memory requirements for the connection-related dynamic arrays. When `active_conx_only =.TRUE.`, the simulation stops once the active connections (involving only the elements defined in the element list) are determined. The active connections are stored in a new file called **`Active_Connection_File`**. Then the simulation can be run using the new connection list, thus having much lower memory requirements. For a thorough discussion of elements and connections in TOUGH+, see Section 7.

`boundaries_in_matrix` Logical variable indicating how inactive elements (describing constant-conditions boundaries, see Section 8.5, and denoted either by a negative or very large volume, or by setting the variable `elem_activity = 'I '`, see Section 7.2) are to be treated when solving the equations of mass and enrgy balance.

When `boundaries_in_matrix =.TRUE.`, then all inactive elements are assigned very large volumes (10^{50} m^3) to maintain constant conditions during the simulation, and are included in the Jacobian matrix. Otherwise, only the active elements are included in the Jacobian.

This feature is useful when using older TOUGH2 input files, and when a parallel version of the code is employed. With newly–developed input files, the `boundaries_in_matrix =.FALSE.` option is highly recommended.

Record `MEMORY.5`

The integer variable `Max_NumSS` (declaring the maximum number of expected sources and sinks) is read using a free format.

Record `MEMORY.6`

In this record, the integer variable `Max_NumMedia` is read using a free format. This variable represents the maximum number of geologic media with different properties to be considered in the simulations.

Record `MEMORY.7`

In `MEMORY.7`, the following logical variables are read using a free format:

```
element_by_element_properties,
   porosity_perm_dependence,
   scaled_capillary_pressure
```

These parameters are defined as follows:

`element_by_element_properties` Logical variable (flag) indicating whether each gridblock has its own hydraulic properties (ϕ and k), in which case they are read on an element-by-element basis (see Sections 7 and 8). This feature is necessary (a) in the simulation of very heterogeneous systems, and/or (b) when k changes in response to pressure and/or temperature variations, or in response to changes in the geomechanical regime of the system.

When `element_by_element_properties = .FALSE.`, then the ϕ and k of a particular element are determined from the general properties of the corresponding porous medium (see Section 6). When `element_by_element_properties = .TRUE.`, then element-specific ϕ and k are read as part of the initial conditions in the **`INCON`** data block (see Section 8).

`porosity_perm_dependence` Logical variable (flag) indicating whether the intrinsic permeability k in a given element is to change as a function of changing porosity ϕ. As discussed earlier, ϕ can change in response to P and/or T changes that are determined from simple or complex geomechanical models (see Section 2).

When `porosity_perm_dependence = .FALSE.`, then k is unaffected by changes in ϕ. When`porosity_perm_dependence = .TRUE.`, then k is readjusted internally to reflect the effect of changes in ϕ that are estimated using either an empirical model (see Equation 2.51, Section 2) or a full geomechanical model.

Note that when `porosity_perm_dependence = .TRUE.`, the variable `element_by_element_properties` is set internally to `.TRUE.` because activation of the `porosity_perm_dependence` feature results in element-specific hydraulic properties.

`scaled_capillary_pressure`Logical variable (flag) indicating whether the capillary pressure P_{cap} will be scaled to reflect variations in ϕ and k.

Activation of this feature by setting `scaled_capillary_pressure = .TRUE.` may be needed (a) in highly heterogeneous systems in which the element-specific properties vary significantly from those described by the average (expected) values of the porous medium (as specified in Section 6), and/or (b) when the significant variations in ϕ and k are experienced in the course of the simulation (e.g., when `porosity_perm_dependence = .TRUE.`).

Record MEMORY.8

In MEMORY.8, the following logical variables are read using a free format:

```
element_by_element_properties,
porosity_perm_dependence,
scaled_capillary_pressure
```

These parameters are defined as follows:

`coupled_geochemistry` Logical variable (flag) indicating whether the simulation involves coupled flow, thermal and geochemical processes. This feature is activated by setting `coupled_geochemistry` to `.TRUE.` when geochemical processes are considered in a TOUGH+ simulation. Because it is not possible to invoke such geochemical processes, `coupled_geochemistry` is set internally to `.FALSE.` in **T+H** simulations.

`property_update` Character variable (flag) indicating the type of property update when coupled geochemical processes are involved in TOUGH+ simulations. Because of the reasons discussed above, it is not used in **T+H** simulations.

Record MEMORY.9

In MEMORY.9, the following logical variables are read using a free format:

```
coupled_geomechanics, geomechanical_code_name,
      property_update, num_geomech_param
```

These parameters are defined as follows:

`coupled_geomechanics` Logical variable (flag) indicating whether the simulation involves coupled flow, thermal and geomechanical processes. Activation of this feature by setting `coupled_geomechanics =.TRUE.` indicates the use of a complex geomechanical model to describe the relationship between hydraulic media properties (ϕ and k) and geomechanical parameters (such as stresses and strains). This complex geomechanical model

overrides the simplified models (based on pore compressibility and expansivity) that are standard in TOUGH+ (see Section 2).

When this flag is invoked, the geomechanicl model invoked by the TOUGH+ code is that defined by the variable `geomechanical_code_name` (see discussion of the next parameter). When `coupled_geomechanics =.FALSE.`, the simplified geomechanical models are invoked, even if the executable of a geomechanical code (named `geomechanical_code_name`) is present in the TOUGH+ directory.

When `coupled_geomechanics =.TRUE.`, and the executable of a geomechanical code named `geomechanical_code_name` is present in the TOUGH+ directory, the following files are created: (a) **`To_GMech`**, containing the data (pressure, temperatures and phase saturations) that are provided to the **`FLAC3D`** geomechanical code from the TOUGH+ simulator for use in the computation of the geomechanical properties of the system, and (b) the **`Fr_Gmech`** file, containing the data (mainly stresses and strains) supplied by the **`FLAC3D`** code for use by the T+H simulator for the computation of the variable (geomechanically-dependent) hydraulic properties ϕ and k. These two files are necessary for communication between the two codes, because the lack of shared memory (disallowed because of intellectual property concerns) makes data exchange by means of these two external files as the only viable option. Additionally, the new file **`Init_Stress`** that stores the initial stresses and strains is created if this is a new run; in a continuation run, the initial (i.e., at t = 0, not at the time of the initiation of the continuation run) stresses and strains are read from the old file **`In_Stresses`** .

`geomechanical_code_name` A character variable of maximum length 6 that provides the name of the executable of the geomechanical code that is coupled with the TOUGH+ code when `coupled_geomechanics =.TRUE.`. In the current implementation of the code, the geomechanical model used in conjunction

with **T+H** is the **FLAC3D** commercial simulator (ITASCA, 2006), i.e., geomechanical_code_name = **FLAC3D**. However, the TOUGH+ can easily accommodate any other geomechanical simulator that conformd with its data exchange formats and requirements.

When coupled_geomechanics =.TRUE. and the executable of a geomechanical code named geomechanical_code_name is present in the TOUGH+ directory, it is treated as a C subroutine that is called by the TOUGH+ simulator. If (a) coupled_geomechanics =.TRUE. and (b) the geomechanical_code_name is different from **FLAC3D** , or does not correspond to any executable available in the TOUGH+ directory, or is blank, the TOUGH+ code resets coupled_geomechanics to =.FALSE..

property_update

Character variable (flag) denoting the manner of property update as a result of interdependent changes in the hydraulic (flow) and geomechanical properties. The variable property_update can assume the following values: 'Continuous', indicating continuous property update (i.e., in every Newtonian iteration of every timestep) and participation in the Jacobian matrix; 'Iteration', indicating property updates in every Newtonian iteration of every timestep, but without any contribution to the Jacobian; and 'Timestep', indicating a single property update at the end of each timestep and no contribution to the Jacobian. The option property_update = 'Continuous' yields the most accurate solutions that are accurate over any pressure range and media geomechanical property range but results in longer execution times, while the option property_update = 'Timestep' leads to faster solutions, but which are acceptably accurate in less compressible media and for mild pressure drops.

If a property_ update value other than the three decribed above is read, then an error message is printed and the simulation is aborted.

`NumGeomechParam` Integer variable defining the number of geomechanical parameters that are to be provided by the geomechanical code (to be read by the TOUGH+ simulator from the **`Fr_Gmech`** file), and which will be used to estimate the updated ϕ and k. Geomechanical codes such as **`FLAC3D`** can provide a very large number of geomechanical properties and parameters of interest, but in the current version of the **T+H** only two (stress and strain) are needed, and these are stored separetely. Thus, `NumGeomechParam` is not needed and can be set to zero (indicating no additional memory utilization) when the **`FLAC3D`** geomechanical model is invoked for hydrate simulations that involve coupled geomechanical effects. However, the parameter is available for generality and maximum flexibility in future code developments that may include the use of additional geomechanical simulators and constitutive equations.

6.0. Physical Properties of System

In this section, the physical properties of the system, including hydrate properties (block **HYDRATE**), rock properties (block **ROCKS** or **MEDIA**), relative permeability and capillary pressure properties (block **RPCAP**), properties specifying multi-component diffusion (block **DIFFU**) are described, followed by the description of a procedure for introducing block-by-block permeability modification.

6.1. Data Block HYDRATE

The parameters describing the hydrate properties and behavior are provided here. Note that free format is used to read the data in this data block.

Record `HYDRATE.1`

The number of component hydrates `NCom` is read in this card. Because of the dearth of quantitative information on the behavior of complex hydrates (as explained in Section 1.2), `NCom = 1` in this version of **T+H**.

Record `HYDRATE.2`

This card reads the following data:

`nameG`	The name of the hydrate-forming gas
`hydrN`	The corresponding hydration number – see Equation (2.1)
`moleF`	The mole fraction in the composite hydrate (for pure hydrates, `moleF = 1`)

Record `HYDRATE.3`

This record includes `N_ThC`, which is the number of coefficients of the polynomial describing the dependence of the thermal conductivity K_H of the hydrate on temperature T.

Record `HYDRATE.4`

This card includes the coefficients A_n (n =0,...,N_k , N_k =`N_ThC-1`)of the thermal conductivity polynomial

$$K_H = A_0 + A_1T + A_2T^2 + ... + A_nT^n \tag{6.1}$$

Record `HYDRATE.5`

This record includes `N_SpH`, which is the number of coefficients of the polynomial describing the dependence of the specific heat C_H of the hydrate on temperature T.

Record `HYDRATE.6`

This card provides the coefficients B_n (n =0,...,N_c, N_c = `N_SpH-1`) of the specific heat polynomial

$$C_H = B_0 + B_1T + B_2T^2 + ... + B_nT^n \tag{6.2}$$

Record `HYDRATE.7`

This record includes `N_Rho`, which is the number of coefficients of the polynomial describing the dependence of the hydrate density ρ_H on temperature T.
Record `HYDRATE.8`

This card provides the coefficients D_n (n = 0,..., N_c, N_c = `N_Rho - 1`) of the hydrate density polynomial.

$$\rho_H = D_0 + D_1 T + D_2 T^2 + \ldots + D_n T^n \tag{6.3}$$

Record `HYDRATE.9`

This card reads the following inhibitor-related data:

`inhibitor_flag`	Logical variable, (flag) indicating presence of an inhibitor when `inhibitor_flag = .TRUE.`. Note that no additional variable values are read past this point if `inhibitor_flag = .FALSE.`.
`Max_TShift`	Real variable describing the inhibitor-induced reference temperature depression – see Equation (2.36). When salt (NaCl) is the inhibitor in question, `Max_Tshift = 2.0E0`.
`Y_atMax_TShift`	Real variable describing the reference inhibitor mole fraction in the aqueous phase, corresponding to `Max_TShift` – see Equation (2.36). For NaCl, `Y_atMax_Tshift = 1.335E-2`.
`InhibitorMW`	Real variable describing the molecular weight of the inhibitor [g mol^{-1}]. For NaCl, `InhibitorMW = 5.8448E1` g mol^{-1}.
`InhibitorDens`	Real variable describinig the inhibitor density [kg m^{-3}]. For NaCl, `InhibitorDens = 2.6E3` kg m^{-3}.
`InhibitorEnthSol`	Real variable describinig the specific enthalpy of the inhibitor dissolution in water [J kg^{-1}]. For NaCl, `InhibitorEnthSol = 6.6479E4` J kg^{-1}.
`InhibitorCpCoeff`	Real array of dimension 3, containing the coefficients of the inhibitor specific heat C_i vs. T quadratic equation $C_i = C_0 + C_1 T + C_2 T^2$. For NaCl, `InhibitorCpCoeff(1,2,3) = 41.293E0` [J kg^{-1} K^{-1}], `3.3607E-2` [J kg^{-1} K^{-2}], and `-1.3927E-5` [J kg^{-1} K^{-3}].

Record HYDRATE.10

The integer variable EquationOption in Card HYDRATE.10 is an option for the selection of the equation describing the *Pe* vs. *Te* and ΔH^0 vs. *Te* relationships (see discussion in Section 2.9). For EquationOption = 0, the modified equations of *Moridis* [2003] are used. For EquationOption = 1, the equations of *Kamath* [1984] are employed. For EquationOption = 2, the *Pe* vs. *Te* relationship is computed using the equation of *Moridis* [2003], while the ΔH^0 vs. *Te* relationship is obtained from the *Kamath* [1984] equation. A value of EquationOption = 2 is the preferred option.

Record HYDRATE.11

The character variable Reaction_Type (LEN = 5) in Card HYDRATE.11 describes the type of hydrate reaction, and can take one of two values. For simulations under equilibrium conditions, Reaction_Type = 'EQUILIBRIUM'. Kinetic hydrate reactions are considered when Reaction_Type = 'KINETIC'.

Record HYDRATE.12

The card HYDRATE.12 is read only if Reaction_Type = 'KINETIC', and includes the following kinetic dissociation parameters:

ActivationEnergy	Real variable describing the activation energy ΔE_a for the hydrate dissociation [J mol^{-1}] – see Equation (2.6)
IntrinsicRateConstant	Real variable describing the intrinsic hydration reaction constant K_0 [kg m^{-2} Pa^{-1} s^{-1}]
Area_Factor	Real variable describing the area adjustment factor [dimensionless] – see Equation (2.6)

6.2. Data Block ROCKS or MEDIA

This block introduces material parameters for any number of different geologic media (porous or fractured).

Record `ROCKS.1`

Format (A5, I5, 8E10.4)

```
mediaName, NAD, mediaDensG,
mediaPoros, (mediaPerm(i), i = 1,3),
mediaKThrW, mediaSpcHt, PoMedRGrain
```

`mediaName` Material name (rock type).

`NAD` Integer variable; if zero or negative, defaults will take effect for a number of parameters (see below);

≥ 1: will read another data record to override defaults.

≥ 2 and < 5: will read two more records with domain-specific parameters for relative permeability and capillary pressure functions.

= 5: In addition to the three records read for NAD > 2, an additional record will be read with the coefficients of the porosity polynomial $\phi/\phi = F_0 + F_1\Delta P + F_2\Delta P^2 + \ldots + F_n\Delta P^n$, where ϕ is the reference (initial) default porosity and $\Delta P = P\text{-}P_0$ is the deviation from the initial pressure P_0. This equation will be used instead of Equation (2.47) to estimate the effect of pressure on the medium porosity.

= 6: In addition to the three records read for NAD > 2, an additional record will be read with the coefficients of Equations (2.56) and (2.57) that describe the compressibility of an unconsolidated porous medium in the presence of cementing solid phases (such as ice and/or hydrates).

`mediaDensG` Rock grain density [kg/m^3]

`mediaPoros` Default porosity ϕ (void fraction) for all elements belonging to domain `MediumName` for which no other porosity has been specified in block **`INCON`**. Option "**`START`**" is necessary for using the default porosity.

`mediaPerm(i),i=1,…,3` Absolute permeabilities along the three principal axes, as specified by `ConxKi` in block **`CONNE`**.

`mediaKThrW` Formation heat conductivity under fully liquid-saturated conditions [W/m/°C].

`mediaSpcHt` Rock grain specific heat [J/kg/°C].

`PoMedRGrain` Rock grain radius [m]. This is needed for the estimation of the hydrate surface reaction area when kinetic hydrate reactions are invoked. If `PoMedRGrain = 0.0e0` (e.g., when no value is provided), the TOUGH+ code provides a grain radius estimate using the Kozeny-Carman approximation (see Section 2.4).

Record `ROCKS.1.1` (optional, when `NAD` ≥ 1 only)

Format (11E10.4)

```
mediaCompr, mediaExpan, mediaKThrD,
mediaTortu, mediaKlink, mediaOrgCF,
mediaCritSat, mediaPermExpon, mediaBeta,
mediaGama, PhiZeroStress
```

`mediaCompr` Pore compressibility $\alpha_P = (1/\phi)(\partial\phi/\partial P)_T$ [Pa^{-1}], see Equation (2.47) – default $\alpha_P = 0.0$.

`mediaExpan` Pore expansivity $\alpha_T = (1/\phi)(\partial\phi/\partial T)_P$ [1/ °C], see Equation (2.47) – default $\alpha_T = 0.0$.

`mediaKThrD` Formation heat conductivity under desaturated conditions [W m^{-1}K^{-1}] – default is `mediaKThrD` = `mediaKThrW`.

`mediaTortu` Tortuosity factor for binary diffusion. If `mediaTortu` = `0`, a porosity and saturation-dependent tortuosity will be calculated internally from the *Millington and Quirk* [1961] model, Equation (2.53).

`mediaKlink` Klinkenberg parameter b [Pa^{-1}] for enhancing gas phase permeability according to the relationship $k_G = k_L(1 + b/P)$, where k_G and k_L are the effective

permeabilities of the gas and of the liquid (i.e., the aqueous phase in the case of gas hydrates) phase, respectively (default is 0).

`mediaOrgCF` Not used for hydrate simulations.

`mediaCritSat` Critical mobile phase saturation ϕ_c, i.e., mobile phase saturation (= S_A+S_G) at which the permeability of hydrate and/or ice-bearing medium becomes equal to zero – needed only when the EPM model is invoked (see Sections 2.11 and 2.12).

`mediaPermExpon` Permeability reduction exponent for solid phase-bearing systems – See Sections 2.11 and 2.12, Equation (2.40).

`mediaBeta` The parameter β used for the computation of porosity as a function of geomechanical stresses σ according to the equation:

$$\phi = \phi_0 - \phi_{\sigma=0} \{ \exp[\beta(\sigma - P)] - \exp[\beta(\sigma_0 - P_0)] \}$$

where $\phi_{\sigma=0}$ is the medium porosity at zero stress, and the subscript *0* indicates initial conditions.

NOTE: This ϕ computational option can be invoked only when all of the following conditions are met: (a) the option `coupled_geomechanics =.TRUE.` is activated, i.e., when the TOUGH+ simulations are coupled with a full geomechanical model such as the **`FLAC3D`** code, (b) at least one medium $\beta \neq 0$, and (c) all media pore compressibilities $\alpha_P = 0$.

If `coupled_geomechanics =.FALSE.`, this parameter is ignored and the variations in ϕ are computed from one of the other computational options available in TOUGH+. The same occurs when `coupled_geomechanics =.TRUE.`, all media $\beta = 0$, and at the pore compressibility of at least one medium $\alpha_P \neq 0$.

`mediaGama` The parameter γ used for the computation of intrinsic permeability k as an empirical function of variations in the porosity ϕ – See Sections 2.12.2.1, Equation (2.51).

`PhiZeroStress` The porosity at zero stress $\phi_{\sigma=0}$. This variable is used only when the option `coupled_geomechanics =.TRUE.` is activated, i.e., when the TOUGH+ simulations are coupled with a full geomechanical model such as the **FLAC3D** code. If `coupled_geomechanics =.FALSE.`, this variable is ignored.

Record `ROCKS.1.2` (optional, `NAD` ≥ 2 only)

Format (I5, 5X, 7E10.4)

```
RelPermEquationNum,
(RelPermParam(i),i= 1,…,7)
```

`RelPermEquationNum` Integer parameter indicating the type of the relative permeability function of the medium under consideration (see detailed discussion in Section 6.3.1).

`RelPermParam(i),i=1,…,7` Real parameters corresponding to the relative permeability function described by the `RelPermEquationNum` option (see Section 6.3.1).

Record `ROCKS.1.3` (optional, NAD ≥ 2 only)

Format (I5, 5X, 7E10.4)

```
PcapEquationNum, (PcapParam(i),i=1,…7)
```

`PcapEquationNum` Integer parameter indicating the type of the capillary pressure function of the medium under consideration (see detailed discussion in Section 6.3.2).

`PcapParam(i),i=1,…7` Real parameters corresponding to the capillary pressure function described by the `PcapEquationNum` option (see Section 6.3.2).

Record `ROCKS.1.4` (optional, NAD = 5 only, to be used when the media porosity is described as a polynomial function of the pressure change ΔP)

Format (I5, 5X, 7E20.13)

```
PhiPolyOrder, (PhiCoeff (i),i=0,…,6)
```

`PhiPolyOrder`	Order n of the polynomial $\phi/\phi = F_0 + F_1\Delta P + F_2\Delta P^2 + \ldots + F_n\Delta P^n$. For a constant ϕ, `PhiPolyOrder = 0.` NOTE: ΔP is in MPa, not the usual Pa units of pressure.
`PhiCoeff(i),i=0,…,6`	Coefficients F_n (n = 0,…,`PhiPolyOrder`) of the $\phi = \phi(\Delta P)$ polynomial. NOTE: The units of F_n must reflect the fact that ΔP is in MPa.

<u>Record `ROCKS.1.4`</u> (optional, NAD = 6 only, to be used when cementing solid phases such as ice and/or hydrates are present in the pores of unconsolidated media – see Section 2.11.3)

Format (10E10.4)

```
LoComp, SatAtLoComp,
HiComp, SatAtHiComp, DeltaSat
```

`LoComp`	The lower limit of the medium compressibility α_{PL} [Pa^{-1}], corresponding to the full stiffening/strengthening effect of the presence of cementing solid phases such as ice and/or hydrates – see Equation (2.56).
`SatAtLoComp`	= S_{Smax}, i.e., the lowest S_S saturation at which $\alpha_P = \alpha_{PL}$ – see Equations (2.56) and (2.57).
`HiComp`	The upper limit of the medium compressibility α_{PU} [Pa^{-1}], corresponding to the absence of cementing solid phases– see Equation (2.56).
`SatAtHiComp`	= S_{Smin}, i.e., the largest S_S saturation at which $\alpha_P = \alpha_{PU}$ – see Equations (2.56) and (2.57).
`DeltaSat`	The smoothing factor δ – see Equation (2.57). A value of δ = 0.015 is suggested – see Figure 2.10.

Repeat records `ROCKS.1`, `ROCKS.1.1`, `ROCKS.1.2`, `ROCKS.1.3` and `ROCKS.1.4` for all the porous/fractured media in the domain under investigation.

Record `ROCKS.2`
A blank record closes the ROCKS data block.

Note: *The number of media described in* **`ROCKS`**/**`MEDIA`** *cannot exceed the number* `Max_NumMedia` *specified in the* `MEMORY` *data block (See Section 5.1). If this happens, an error message is printed and the simulation is aborted.*

6.3. Data Block RPCAP

This block introduces information on relative permeability and capillary pressure functions, which will be applied for all flow domains for which no data were specified in records `ROCKS.1.2` and `ROCKS.1.3`. A catalog of relative permeability and capillary pressure functions is presented in Sections 6.3.1 and 6.3.2, respectively.

Record `RPCAP.1`
Format (I5, 5X, 7E10.4)
`DefaultRelPermType, (RPD(i),i=1,7)`

`DefaultRelPermType`	Integer parameter describing the type of the default relative permeability function (see Section 6.3.1).
`(RPD(i),I=1,…,7)`	Real parameters corresponding to the relative permeability function selected by the `DefaultRelPermType` parameter (see Section 6.3.1).

Record `RPCAP.2`
Format (I5, 5X, 7E10.4)
`DefaultCapPresType, (CPD(I),i=1,7)`

`DefaultCapPresType`	Integer parameter describing the type of the default capillary pressure function (see Section 6.3.2).

`(CPD(i), i=1,…,7)` Real parameters corresponding to the capillary pressure function selected by the `DefaultCapPresType` parameter (see Section 6.3.2).

6.3.1. Relative Permeability Functions

6.3.1.1. `RelPermEquationNum = 1:` **Linear functions**

k_{rA} increases linearly from 0 to 1 in the range `RP(1)` $\le S_A \le$ `RP(3)`;

k_{rG} increases linearly from 0 to 1 in the range `RP(2)` $\le S_G \le$ `RP(4)`

Restrictions: `RP(3) > RP(1);  RP(4) > RP(2).`

6.3.1.2. `RelPermEquationNum = 2:` **Power functions**

$k_{rA} = (S_A)^n$, $k_{rG} = 1$, where n = `RP(1)`

6.3.1.3. `RelPermEquationNum = 3:` **Corey's curves** [*Corey*,1954]

$$k_{rA} = \hat{S}^4, \qquad k_{rG} = \left(1-\hat{S}\right)^2 \left(1-\hat{S}^2\right)$$

where $$\hat{S} = \frac{\left(S_A - S_{irA}\right)}{\left(1 - S_{irA} - S_{irG}\right)}$$

with S_{irA} = `RP(1);` S_{irG} = `RP(2)`

Restrictions: `RP(1) + RP(2) < 1`

6.3.1.4. `RelPermEquationNum = 4:` **Grant's curves** [*Grant*, 1977]

$$k_{rA} = \hat{S}^4, \qquad k_{rG} = 1 - k_{rA}$$

where $$\hat{S} = \frac{\left(S_A - S_{irA}\right)}{\left(1 - S_{irA} - S_{irG}\right)}$$

with S_{irA} = `RP(1);` S_{irG} = `RP(2)`

Restrictions: `RP(1) + RP(2) < 1`

6.3.1.5. `RelPermEquationNum = 5:` **All phases perfectly mobile**

$k_{rG} = k_{rA} = 1$ for all saturations; no parameters

6.3.1.6. `RelPermEquationNum = 6:` **Functions of Fatt and Klikoff** (1959)

$$k_{rA} = \left(S^*\right)^3, \qquad k_{rG} = \left(1 - S^*\right)^3$$

where: $$S^* = \frac{\left(S_A - S_{irA}\right)}{\left(1 - S_{irA}\right)}$$

with S_{irA} = `RP(1)`.

Restriction: `RP(1) < 1`.

6.3.1.7. `RelPermEquationNum = 7, -7:` **van Genuchten-Mualem model** [*Mualem*, 1976; *van Genuchten*, 1980]

$$k_{rA} = \begin{cases} \sqrt{S^*}\left\{1 - \left(1 - [S^*]^{1/\lambda}\right)^{\lambda}\right\}^2 & \text{if } S_A < S_{mxA} \\ 1 & \text{if } S_A \geq S_{mxA} \end{cases}$$

Gas relative permeability can be chosen from among several options. For `RelPermEquationNum = 7`, it is computed from one of the following three forms, of which the second is the *Corey* [1954] equation and the third is the modified Stone equation [*Stone*, 1970] – see Section 6.1.3.9:

$$k_{rG} = \begin{cases} 1 - k_{rG} & \text{if } S_{irG} = 0 \\ \left(1 - \hat{S}\right)^2 \left(1 - \hat{S}^2\right) & \text{if } S_{irG} > 0 \text{ and } n_G \leq 0 \\ \left(\tilde{S}\right)^{n_G} & \text{if } S_{irG} > 0 \text{ and } n_G > 0 \end{cases}$$

For `RelPermEquationNum =-7`, the gas relative permeability is computed from the model of *Parker et al.* [1987]:

$$k_{rG} = \begin{cases} \sqrt{1 - S^*}\left\{\left[1 - \left(S^*\right)^{1/\lambda}\right]^{\lambda}\right\}^2 & \text{if } 1 - S^* > 0 \\ 0 & \text{if } 1 - S^* \leq 0 \end{cases}$$

The k_{rA} and k_{rG} estimates are subject to the following restrictions:

$$0 \leq k_{rA}, k_{rG} \leq 1$$

Here, $S^* = \dfrac{S_A - S_{irA}}{S_{mxA} - S_{irA}}$, $\hat{S} = \dfrac{S_A - S_{irA}}{1 - S_{irA} - S_{irG}}$ $\tilde{S} = \dfrac{S_G - S_{irG}}{1 - S_{irA}}$

Parameters: `RP(1)` $= \lambda$
`RP(2)` $= S_{irA}$
`RP(3)` $= S_{mxA}$
`RP(4)` $= S_{irG}$
`RP(5)` $= n_G$

Notation: Parameter λ is m in van Genuchten's notation, with $m = 1 - 1/n$.

6.3.1.8. `RelPermEquationNum = 8:` **Function of Verma et al.** [1985]

$$k_{rA} = \hat{S}^3, \qquad k_{rG} = A + B\hat{S} + C\hat{S}^2$$

where $$\hat{S} = \frac{(S_A - S_{irA})}{(S_{mxA} - S_{irA})}$$

Parameters as measured by *Verma et al*. [1985] for steam-water flow in an unconsolidated sand:

S_{irA}	=	`RP(1)`	= 0.2
S_{mxA}	=	`RP(2)`	= 0.895
A	=	`RP(3)`	= 1.259
B	=	`RP(4)`	= -1.7615
C	=	`RP(5)`	= 0.5089

6.3.1.9. `RelPermEquationNum = 9:` Modified version of Stone's first three-phase relative permeability method [*Stone*, 1970].

$$k_{rA} = \max\left\{0, \min\left\{\left[\frac{S_A - S_{irA}}{1 - S_{irA}}\right]^n, 1\right\}\right\},$$

$$k_{rG} = \max\left\{0, \min\left\{\left[\frac{S_G - S_{irG}}{1 - S_{irA}}\right]^{n_G}, 1\right\}\right\},$$

$$k_{rH} = 0$$

Parameters are S_{irA} = `RP(1)`, S_{irG} = `RP(2)`, n = `RP(3)`, n_G = `RP(4)`.

Note: When `RP(4)=0.0e0`, it is reset internally to `RP(4)=RP(3)` .

The user may modify the source code of TOUGH+ (segment **`T_Media_Properties.f95`**) to employ other relative permeability relationships. However, for this task the user needs to be familiar with the concepts of generic processes (and overloading) in object oriented programming languages.

6.3.2. Capillary Pressure Functions

6.3.2.1. `PcapEquationNum = 1:` **Linear function**

$$P_{cap} = \begin{cases} -\mathrm{CP}(1) & \text{for } S_A \le \mathrm{CP}(2) \\ 0 & \text{for } S_A \le \mathrm{CP}(2) \\ -\mathrm{CP}(1)\dfrac{\mathrm{CP}(3)-S_A}{\mathrm{CP}(3)-\mathrm{CP}(2)} & \text{for } \mathrm{CP}(2)<S_A<\mathrm{CP}(3) \end{cases}$$

Restriction: `CP(3)` > `CP(2)`

6.3.2.2. `PcapEquationNum = 2:` **Function of Pickens et al.** [1979]

$$P_{cap} = -P_0 \left\{ \ln\left[\frac{A}{B}\left(1+\sqrt{1-B^2/A^2}\right)\right]\right\}^{1/x}$$

with

$$A = \left(1+\frac{S_A}{S_{A0}}\right)\frac{\left(S_{A0}-S_{irA}\right)}{\left(S_{A0}+S_{irA}\right)}, \qquad B = 1-\frac{S_A}{S_{A0}}$$

where P_0 = `CP(1)`, S_{irA} = `CP(2)`, S_{A0} = `CP(3)`, x = `CP(4)`

Restrictions: `0 < CP(2) < 1` • `CP(3)`; `CP(4)` $\neq$ `0`

6.3.2.3. `PcapEquationNum = 3:` **TRUST capillary pressure** [*Narasimhan et al.*, 1978]

$$P_{cap} = \begin{cases} -P_e - P_0\left[\dfrac{1-S_A}{S_A - S_{irA}}\right]^{1/\eta} & \text{for } S_A < 1 \\ 0 & \text{for } S_A < 1 \end{cases}$$

where P_0 = `CP(1)`, S_{irA} = `CP(2)`, η = `CP(3)`, P_e = `CP(4)`

Restrictions: `CP(2)` $\geq$ `0`; `CP(3)` $\neq$ `0`

6.3.2.4. `PcapEquationNum` **= 4 : Milly's function** [*Milly*, 1982]

$$P_{cap} = -97.783 \times 10^{A}, \quad A = 2.26\left(\frac{0.371}{S_A - S_{irA}} - 1\right)^{1/4}$$

where S_{irA} = `CP(1)`

Restriction: `CP(1)` $\geq$ `0`

6.3.2.5. `PcapEquationNum` **= 6 : Leverett's function** [*Leverett*, 1941; *Udell and Fitch*, 1985]

$$P_{cap} = -P_0 \cdot \sigma(T) \cdot f(S_A)$$

where

$\sigma(T)$: surface tension of water (supplied internally in TOUGH+)

$$f(S_A) = 1.417\left(1-S^*\right) - 2.120\left(1-S^*\right)^2 + 1.263\left(1-S^*\right)^3$$

$$S^* = \frac{\left(S_A - S_{irA}\right)}{\left(1 - S_{irA}\right)}$$

Parameters: P_0 = `CP(1)`, S_{irA} = `CP(2)`

Restriction: `0` $\leq$ `CP(2)` $<$ `1`

6.3.2.6. `PcapEquationNum` **= 7 : van Genuchten function** [*van Genuchten*, 1980]

$$P_{cap} = -P_0\left[\left(S^*\right)^{-1/\lambda} - 1\right]^{1-\lambda}, \quad S^* = \frac{\left(S_A - S_{irA}\right)}{\left(S_{mxA} - S_{irA}\right)}$$

subject to the restriction: $-P_{\max} \le P_{cap} \le 0$

Parameters: `CP(1)` $= \lambda = 1 - 1/n$

`CP(2)` $= S_{irA}$ (should be chosen smaller than the corresponding parameter in the relative permeability function; see note below.)

`CP(3)` $= 1/P_0 = \alpha/\rho_w g$

`CP(4)` $= P_{max}$

`CP(5)` $= S_{mxA}$

Notation: Parameter λ is m in van Genuchten's notation, with $m = 1 - 1/n$.

<u>Note on parameter choices:</u> In the van Genuchten's derivation [1980] of the capillary pressure equation, the parameter S_{irA} for irreducible water saturation is the same in the relative permeability and capillary pressure functions. As a consequence, for $S_A \to S_{irA}$ we have $k_{rA} \to 0$ and $P_{cap} \to -\infty$, which is unphysical because it implies that the radii of capillary menisci go to zero as liquid phase is becoming immobile (discontinuous). In reality, no special capillary pressure effects are expected when liquid phase becomes discontinuous. Accordingly, we recommend to always define a smaller S_{irA} for the capillary pressure as compared to that for the relative permeability function.

6.3.2.7. `PcapEquationNum = 8:` Brooks-Corey equation modified to account for effect of hydrate on capillary pressure

$$P_{cap} = -F \cdot G \cdot P_{GE}(S^*)^v, \; S^* = \frac{\left(S_A - S_{irA}\right)}{\left(1 - S_{irA}\right)}, \; F = 1 + A \cdot Bx(a, b, S_H)$$

where

v = exponent with the following restrictions: v< 0 and |v| ≤ 1;
P_{GE} = gas entry pressure;
G = error function equation that smoothes curve near S=0;

F = factor that describes effect of hydrate on capillary pressure;
A = parameter > 0;
B_x = incomplete beta function;
a,b = input arguments for B_x;
S_H = hydrate saturation.

Parameters: `CP(1)` = P_{GE}, `CP(2)` = v, `CP(3)` = S_{irA},
`CP(4)` = $P_{cap,max}$, `CP(5)` = A, `CP(6)` = a, `CP(7)` = b

6.3.2.8. `PcapEquationNum = 9:` **No capillary pressure**

$P_{cap} \equiv 0$ for all saturations; no parameters.

6.4. Data Block `DIFFUSION`

This block reads multicomponent diffusion coefficients using a `NAMELIST` format. This is a very powerful format that allows maximum clarity and flexibility, accepting free formats, arbitrary ordering of variables, insertions of comments anywhere in the input fields, and providing the option of ignoring any of the `NAMELIST` parameters by not assigning a value to it. For more information, the reader is directed to a textbook on FORTRAN 95/2003.

Record `DIFFUSION.1`

This record includes general data describing key diffusion parameters. The namelist in this record is named `Diffusion_Key_Parameters`, and includes the following parameters:

`gas_diffusivity_equation_exponent:` A double precision variable describing the dependence of gas diffusivity on temperature (see Equation 6.4). The default value is 1.80.

`P_at_RefDiffusivity:` Pressure at the reference diffusivity (in Pa). If `P_at_RefDiffusivity <= 0`, the default value is 10^5 Pa.

`Tk_at_RefDiffusivity:` Temperature at the reference diffusivity (in K). If `T_at_RefDiffusivity <= 0`, the default value is 273.15 K.

`Option_tortuosity_CompuMethod:` A character variable describing the method of tortuosity estimation. The possible values are '`Saturation`', '`Relative_Perm`' and '`Constant`' (see discussion in Section 6.4.1).

`Option_gas_diffusivity_CompuMethod:` A character variable describing the method of estimation of the binary **gas** diffusivities. The following options are available:

`= 'Standard'` : This option involves the application of Equation (6.4), and requires non-zero multicomponent gas diffusivity values read from the standard input file.

`= 'Real_Gas_EOS'` : In this case, the binary gas diffusivities are computed from the cubic equation of state used to determine all the real gas properties.

`= 'Constant'` : When this option is invoked, the constant multicomponen diffusivity values provided by the input file are used.

`full_multiphase_diffusion:` A logical variable describing the method of estimation of the method of estimation of multiphase diffusive fluxes. The following options are available:

`= .TRUE.` : With this option, harmonic weighting to the full multiphase effective diffusion strength is applied. This includes contributions from gas and aqueous phases, accounts for coupling of diffusion with phase partitioning effects, and can describe the most general cases of diffusion across phase boundaries.

`= .FALSE.` : In this case, harmonic weighting is performed separately for the diffusive fluxes in the mobile phases.

<u>Records `DIFFUSION.2.1`, `DIFFUSION.2.2`, etc.</u>

Record DIFFUSION.1 is followed by DIFFUSION.2.x records, with x = 1,…,NubMobPhases (i.e., the number of mobile phases in the system under study). These records describe component diffusivities in the various phases. The same namelist is used in each one of these records. It is named Component_Diffusivities_in_Phases, and includes the following parameters:

phase: A character variable identifying the mobile phase for which the diffusivities of the various components are reported. The possible options in the **T+H** code are 'Aqueous' and 'Gas'.

phase_number: An integer variable providing the number of the phase in the phase numbering sequence used in the code. The possible options in the **T+H** code are:
= 2 for phase = 'Aqueous', and
= 1 for phase = 'Gas'.

component: A character array of dimension NumCom (see Section 5.1) identifying the various mass components partioned in the phase in question (denoted by phase). The possible options in the **T+H** code are 'CH4', 'H2O' and 'NaCl'.

component_number: An integer array providing the number of the component in the numbering sequence used in the code. The possible options in the **T+H** code are:
= 1 for component = 'CH4'
= 2 for component = 'H2O'
= 3 for component = 'NaCl' (if present)

component_diffusivity: A double precision array of dimension NumCom (see Section 5.1) describing the value of the multicomponent diffusivities D_β^κ (see Equations (2.59) and (6.4)) of the various components κ in the phase β under consideration (indentified by phase and phase_number, respectively).

NOTE: *The records* DIFFUSION.2.x *must provide data for all mobile phases and all components, even if the gas diffusivities may be overridden internally when* Option_gas_diffusivity_CompuMethod = 'Real_Gas_EOS'.

The structure of the namelists `Diffusion_Key_Parameters` and `Component_Diffusivities_in_Phases` (and their use as input formats in the data block **`DIFFUSION`**) are best illustrated in the example of **Figure 6.1**.

```
DIFFUSION-----*----2----*----3----*----4----*----5----*----6----*----7----*----8
&Diffusion_Key_Parameters  gas_diffusivity_equation_exponent  = 1.8d0
                           P_at_RefDiffusivity                = 1.0d5,    ! in Pa
                           Tk_at_RefDiffusivity               = 273.15d0, ! in K
                           Option_tortuosity_CompuMethod      = 'Saturation',
                           Option_gas_diffusivity_CompuMethod = 'Real_Gas_EOS',
                           full_multiphase_diffusion          = .TRUE.
                           /
&Component Diffusivities in Phases  phase        = 'Aqueous',  phase number = 2,
                                    component(1) = 'CH4',
                                    component_number(1) = 1,
                                    component_diffusivity(1) = 1.0d-10,  ! (m2/s)
                                  ! …. Diffusivity of component 1 in phase 2
                                    component(2) = 'H2O',
                                    component_number(2) = 2,
                                    component_diffusivity(2) = 1.0d-10,  ! (m2/s)
                                  ! …. Diffusivity of component 2 in phase 2
                                    component(3) = 'NaCl',
                                    component_number(3) = 3,
                                    component_diffusivity(3) = 1.0d-1    ! (m2/s)
                                  ! …. Diffusivity of component 3 in phase 2
                                    /
&Component_Diffusivities_in_Phases  phase        = 'Gas',      phase_number = 1,
                                    component(1) = 'CH4',
                                    component_number(1) = 1,
                                    component_diffusivity(1) = 1.0d-5,  ! (m2/s)
                                  ! …. Diffusivity of component 1 in phase 1
                                    component(2) = 'H2O',
                                    component_number(2) = 2,
                                    component_diffusivity(2) = 1.0d-5,  ! (m2/s)
                                  ! …. Diffusivity of component 2 in phase 1
                                    component(3) = 'NaCl',
                                    component_number(3) = 3,
                                    component_diffusivity(3) = 0.0d-1    ! (m2/s)
                                  ! …. Diffusivity of component 3 in phase 1
                                    /
```

Figure 6.1. The **`DIFFUSION`** data block, with examples of the `Diffusion_Key_Parameters` and `Component_Diffusivities_in_Phases` namelists

6.4.1. User Options for Multiphase Diffusion

The treatment of full multiphase diffusion requires parameter specifications in two different data blocks. First, the user must set `binary_diffusion` to a value of `.TRUE.` in data block **`MEMORY`**. Diffusivities are input through the data block **`DIFFU`** (Section 6.4). See note in Section 6.4 regarding an alternative approach, which allows user to only specify the diffusion coefficient of salt and thus save in computation costs.

For two gaseous components, the multicomponent diffusivities in the gas phase are the binary gas diffusivities. For 3 gaseous components κ_1, κ_2 and κ_3, the multicomponent diffusivities of gas component κ_i are computed from the binary gas diffusivities by the Wilke method [API, 1977] as

$$D_G^{\kappa_i} = \frac{1 - Y_G^{\kappa_i}}{\sum_{n=1, n\neq i}^{3} \frac{Y_G^{\kappa_n}}{d_G^{\kappa_i \kappa_n}}} \tag{6.4}$$

Extension to more than three gas components is entirely analogous. When computing binary gas diffusivities using the option `Option_gas_diffusivity_CompuMethod = 'Real_Gas_EOS'`, their pressure and temperature dependence is automatically accounted for in the computations. When `Option_gas_diffusivity_CompuMethod = 'Standard'`, the binary gas diffusivities of any two components κ_1 and κ_2 depend on pressure and temperature as [*Vargaftik*, 1975; *Walker et al.*, 1981]

$$d_G^{\kappa_1 \kappa_2}(P,T) = d_G^{\kappa_1 \kappa_2}(P_0,T_0) \frac{P_0}{P} \left[\frac{T + 273.15}{273.15} \right]^{\theta} \tag{6.5}$$

For example, at standard conditions of P_0 = 1 atm = 1.01325 bar, T_0 = 0 ˚C, the diffusion coefficient for vapor-air mixtures has a value of 2.13×10^{-5} m^2/s; parameter θ for the temperature dependence is 1.80. TOUGH+ can model a temperature dependence of gas

phase diffusion coefficients according to Equation (6.5) by specifying parameter θ = `gas_diffusivity_equation_exponent` in the first record of data block **DIFFUSION** (see Section 6.4). Presently there are no provisions for allowing the parameter θ of temperature dependence on the gas phase composition.

Tortuosity effects have a porous medium-dependent part τ_0 and a saturation-dependent part τ_β, see Equation (2.47). The following three alternative formulations are available, as defined by the `Option_tortuosity_CompuMethod` variable:

1. Relative permeability model: for domains for which a tortuosity parameter τ_0 = `mediaTortu` $\neq 0$ is specified in data block **MEDIA** or **ROCKS**, tortuosity will be taken as $\tau = \tau_0\, \tau_\beta(S_\beta) = \tau_0\, k_{r\beta}(S_\beta)$.
2. Saturation model: for domains in which `mediaTortu` $\neq 0$, tortuosity will be taken as $\tau = \tau_0\, \tau_\beta(S_\beta) = \tau_0\, S_\beta$.
3. Constant tortuosity: the constant `mediaTortu` values provided in the data block **MEDIA** or **ROCKS** (Section 5) will be used.

If `mediaTortu = 0` in data block **MEDIA** or **ROCKS**, then tortuosity is computed from the Millington-Quirk model in Equation (2.21) as $\tau = \tau_0 \tau_\beta = \phi^{1/3} S_\beta^{10/3}$. This value is then used in the computation of the effective diffusion coefficient in Equation (2.60).

6.5. Block-by-Block Permeability Modification

TOUGH+ provides a feature that applies permeability modification (PM) coefficients for individual grid blocks according to

$$k_n \quad \rightarrow \quad k'_n \quad = \quad k_n \cdot \zeta_n \tag{6.6}$$

Here, k_n is the absolute (intrinsic or reference) permeability of grid block (cell, element) n, as specified in data block **ROCKS** or **MEDIA** for the domain to which that grid block belongs, while ζ_n is the permeability modification coefficient. The strength of capillary pressure will be automatically scaled according to *Leverett* [1941] as:

$$P_{cap,n} \quad \rightarrow \quad P'_{cap,n} \quad = \quad \frac{P_{cap,n}}{\sqrt{\zeta_n}} \tag{6.7}$$

The subroutine `Initialize_Perm_Modifiers` is called after all **MESH** data have been processed. The `Initialize_Perm_Modifiers` routine initializes permeability modifiers and generates informative printout. To engage block-by-block permeability modification, users need to specify a (dummy) domain named "`SEED`" in block **ROCKS** or **MEDIA**. No grid blocks should be assigned to this domain; the presence of domain "`SEED`" simply serves as a flag to put permeability modification into effect, while data provided in domain "`SEED`" serve to select different options. Random (spatially uncorrelated) PM data can be internally generated in TOUGH+. Alternatively, externally defined permeability modifiers may be provided as part of the geometry data in block **ELEME**. This feature can be used to apply spatially correlated fields: users can run their own geostatistical package to generate whatever fields they desire, and then use a preprocessing program to place the PM-coefficients into the **ELEME** data block.

Note that this approach of block-by-block permeability modification affects only grid block permeabilities but not porosities. Full details on the various user options for PM-

coefficients are given in informative printout that is automatically generated with each TOUGH+ run.

In addition to this approach, it is possible for the users to apply a geostatistical package of their choice to develop element-specific (block-by-block) statistically heterogeneous porosities ϕ, intrinsic permeabilities k, and phase saturations. These data can then be read as part of the **INCON** data block, as will be discussed in Section 8.

7.0. Geometrical Representation, Domain Discretization, and Grid Generation

The data blocks that specify the geometrical representation of the hydrate system are discussed in this section, including the specification of elements (block **`ELEM`**), connections between elements (block **`CONNE`**), and the generation of grids (block **`MESHM`**) for radial symmetric grids, rectilinear grids, and grids containing fractured media. First, we discuss the convention used in TOUGH+ for entering and processing geometrical data.

7.1. TOUGH+ Convention for Geometrical Data

Handling of flow geometry data in TOUGH+ is upward compatible with TOUGH2 and TOUGH input formats and data handling. As in other *integral finite difference* codes [*Edwards*, 1972; *Narasimhan and Witherspoon*, 1976], flow geometry is defined by

means of a list of volume elements (*grid blocks*), and a list of flow connections between them (Section 3.3). This formulation can handle regular and irregular flow geometries in one, two, and three dimensions. Single- and multiple-porosity systems (porous and fractured media) can be specified, and higher order methods, such as seven- and nine-point differencing, can be implemented by means of appropriate specification of geometric data [*Pruess and Bodvarsson*, 1983].

In TOUGH+, as in TOUGH2, volume elements are identified by names that consist of a string of either five or eight characters, '`12345`' or '`12345678`'. These are arbitrary, except that the last two characters (#4 and #5) must be numbers; an example of a valid 5-character element name is '`ELE10`', and an example of a valid 8-character name is '`AB00CC23`'. Flow connections are specified as ordered pairs of elements, such as ('`ELE10`', '`ELE11`'). A variety of options and facilities are available for entering and processing geometric data. As in TOUGH2, element volumes and domain identification can be provided by means of a data block **`ELEME`** in the INPUT file, while a data block **`CONNE`** can be used to supply connection data, including interface area, nodal distances from the interface, and orientation of the nodal line relative to the vertical (see Sections 7.2 and 7.3). These data are internally written to a disk file **`MESH`**, which in turn initializes the geometry data arrays used during the flow simulation. It is also possible to omit the **`ELEME`** and **`CONNE`** data blocks from the **`INPUT`** file, and provide geometry data directly on a disk file **`MESH`**.

TOUGH+ provides a capability for describing the flow system geometry and discretizing the domain by means of the **`MeshMaker.f95`** application written in a FORTRAN95/2003 (see Section 7.4). Unlike the approach in the TOUGH [*Pruess*,

1987] and TOUGH2 [*Pruess*, 1991a; *Pruess et al*., 1999] codes, **`MeshMaker.f95`** is separate from (i.e., it is not integrated within) the TOUGH+ code. This application can perform a number of mesh generation and processing operations. The **`MeshMaker.f95`** code is written according to the tenets of Object-Oriented Programming, and has a modular structure. It can generate two-dimensional radially symmetric (*r,z*) meshes, and one-, two-, and three-dimensional rectilinear (Cartesian) grids in (*x,y,z*). Multiple-porosity processing for simulation of flow in naturally fractured reservoirs can be invoked by means of a keyword **`MINC`**, which stands for *Multiple INteracting Continua* (see Section 2.14). The `MINC` process operates on the data of the primary (porous medium) mesh as provided on disk file **`MESH`**, and generates a secondary mesh containing fracture and matrix elements with identical data formats on file **`MINC`**. The file **`MESH`** used in this process can be either directly supplied by the user, or it can have been generated from an earlier application of the **`MeshMaker.f95`** application.

7.2. Data Block ELEME

This block introduces element (grid block) information.

<u>Record `ELEME.1` (for 5-character elements only)</u>

Format (A3, I2, 2I5, A5, 6E10.4, 1X, A1)

```
ElName5C, NSEQ, NADD, MA12,
elem_vol, elem_aht, elem_pm,
X, Y, Z, elem_activity
```

`ElName5C`	The five-character name of an element. The first three characters can be letters, numbers or blanks, and the last two characters (`NE = ElName5C(4:5)`) must be numbers.

NSEQ	The number of additional elements having the same volume and belonging to the same reservoir domain.
NADD	The increment between the code numbers of two successive elements. (Note: if the longest dimension in the grid is <1000, the maximum permissible code number NE + NSEQ *NADD is ≤ 999.)
MA12	A five-character material identifier corresponding to one of the reservoir domains as specified in block **ROCKS**. If the first three characters are blanks and the last two characters are numbers, then they indicate the sequence number of the domain as entered in **ROCKS**. If left blank the element is by default assigned to the first domain in block **ROCKS**.
elem_vol	The element volume (m^3).
elem_aht	The interface area (m^2) for heat exchange with semi-infinite confining beds. Internal **MESH** generation via the **MeshMaker.f95** facility will automatically generate elem_aht.
elem_pm	The permeability modifier (optional, active only when a domain "SEED" has been specified in the **ROCKS** block; see Section 6.5). Will be used as multiplicative factor for the permeability parameters from block **ROCKS**. Simultaneously, the capillary pressure strength will be scaled as $elem_pm^{-1/2}$. elem_pm = 0 will result in an impermeable block. Random permeability modifiers can be generated internally, see detailed comments in the TOUGH+ output file. The elem_pm may be used to specify spatially correlated heterogeneous fields, but users need their own preprocessing programs for this, as TOUGH+ provides no internal capabilities for generating such fields.
X, Y, Z	Cartesian coordinates of the grid block centers. These are included in the **ELEME** data for listing in output files of the simulation results that conform to the specifications of the **TecPlot** package. Additionally, the coordinate data are be needed for

(a) the computation of interblock properties in cylindrical grids, (b) the initialization of a gravity-capillary equilibrium (such as the one described in Sections 6.9 and 9.10 of *Pruess et al.* [1999]), and (c) for the computation of quantities related to mechanical dispersion in solute transport investigations [*Moridis et al*., 1999].

`elem_activity` A (optional) character variable that describes the activity status of an element. An element is rendered inactive (i.e., acting as a Dirichlet boundary) by invoking the options `elem_activity = I` (indicating that the element conditions are time-invariant) or `elem_activity = V` (denoting time-variable conditions). The grid block is assumed active by default for any other value of `elem_activity`, including a blank (see Section 8.4).

Record `ELEME.1` (for 8-character elements only)

Format (A8, 7X, A5, 6E10.4, 1X, A1)

```
ElName8C, MA12,
elem_vol, elem_aht, elem_pm,
X, Y, Z, elem_activity
```

`ElName8C` The eight-character name of an element. The first five characters can be letters, numbers or blanks, and the last two characters (`NE = ElName8C(7:8)`) must be numbers.

All other variables and parameters are as previously defined in case of the five-character element names.

Repeat record `ELEME.1` for the number of elements desired.

Record `ELEME.2` A blank record closes the **`ELEME`** data block.

NOTE: *The number of elements described in the* **`ELEME`** *data block cannot exceed the number* `Max_NumElem` *specified in the* `MEMORY` *data block (See Section 5.1). If this happens, an error message is printed and the simulation is aborted.*

7.3. Data Block CONNE

This block introduces information for the connections (interfaces) between elements.

Record CONNE.1 (for connections involving 5-character elements only)

Format (A5, A5, 4I5, 5E10.4)

```
ConxName1, ConxName2, NSEQ, NAD1, NAD2,
ConxKi, ConxD1, ConxD2, ConxArea, ConxBeta,
emissivity
```

ConxName1	The name of the first five-character element in the connection.
ConxName2	The code name of the five-character second element in the connection.
NSEQ	The number of additional connections in the sequence.
NAD1	The increment of the number of the first element (as defined by the last two characters of its name, i.e., ConxName1(4:5)) between two successive connections.
NAD2	The increment of number of the second element (as defined by the last two characters of its name, i.e., ConxName2(4:5)) between two successive connections.
ConxKi	The permeability index of the connection. Setting it equal to 1, 2, or 3 specifies absolute permeability to be MediaPerm(ConxKi) for the materials in elements (ConxName1) and (ConxName2), where MediaPerm is read in block **ROCKS**. This allows assignment of different permeabilities, e.g., in the horizontal and vertical direction.
ConxD1, ConxD2	The distance [m] between the common interface of a connection from the centers of the first and second element in the connection, respectively.
ConxArea	The element interface area [m^2].
ConxBeta	The cosine of the angle between the gravitational acceleration vector and the line between the two

elements. `ConxBeta*gravity > 0 (<0)` corresponds to first element being above (below) the second element.

`emissivity` The *radiant emittance* factor for radiative heat transfer, which for a perfectly "black" body is equal to 1. The rate of radiative heat transfer between the two grid blocks is:

$$Q_{H,rad} = \text{emissivity} * \sigma_0 * \text{ConxArea} * (T_2^4 - T_1^4)$$

where σ_0 = 5.6687e-8 J/m^2 K^4 s is the Stefan-Boltzmann constant, and T_1 and T_2 are the absolute temperatures of the two grid blocks. The term `emissivity` may be entered as a negative number, in which case the absolute value will be used, and heat conduction at the connection will be suppressed. Setting `emissivity = 0.0e0` will result in no radiative heat transfer.

<u>Record `CONNE.1` (for connections involving 5-character elements only)</u>

Format (A8, A8, 9X, I5, 5E10.4)

```
Conx8Name1, Conx8Name2, ConxKi,
ConxD1, ConxD2, ConxArea, ConxBeta,
emissivity
```

`Conx8Name1` The name of the first eight-character element in the connection.

`Conx8Name2` The code name of the second eight-character element in the connection.

All other variables and parameters are as previously defined in case of connections between five-character elements.

Repeat record `CONNE.1` for the number of connections desired.

<u>Record `CONNE.2`</u> A blank record closes the **`CONNE`** data block. Alternatively, connection information may terminate on a record with `'+++  '` typed in the first five columns, followed by element cross-referencing information. The second type of termination is generated in the **`MESH`** file upon completion of a TOUGH+ run.

NOTE: The number of connections described in the **CONNE** *data block cannot exceed the number* `Max_NumConx` *specified in the* `MEMORY` *data block (See Section 5.1). If this happens, an error message is printed and the simulation is aborted*

7.4. The `MeshMaker.f95` Facility

In this section we discuss the use of the **`MeshMaker.f95`** facility, and the required parameter inputs for mesh generation and processing. As indicated earlier, unlike in TOUGH2 [*Pruess et al*., 1999], **`MeshMaker.f95`** is not integrated into the TOUGH+ code but is an independent program that is written in FORTRAN 95/2003, has a modular structure, and an architecture based on the principles of Object-Oriented Programming. The **`MeshMaker.f95`** input has a modular structure, which is organized by keywords in a manner analogous to that of TOUGH+. This section provides detailed instructions for preparing the input files, illustrative examples of which are shown in **Figures 7.1** to **7.4**.

7.4.1. Inputs Related to Problem Definition and Dimensioning

These inputs occupy two records (both mandatory), and provide (a) a short description of the problem through an informative title, and (b) data to allow proper dimensioning of the work arrays and formatting of the **`MESH`** file to be generated. These two initial records are discussed in detail below:

<u>Record `MESHMAKER.1`</u>

The first record of the input file in any **`MeshMaker.f95`** application is the character variable **`TITLE`**, which includes a header of up to 80 characters and is

read using a free format. This record is necessary for any **MeshMaker** simulation to begin.

Record MESHMAKER.2

The following variables are read in MESHMAKER.2 using a free format:

```
MaxNum_Elem, Longest, ElemNameLength, FormatType
```

These parameters are defined as follows:

Max_NumElem	Integer denoting the maximum number of elements in the grid under construction.
Longest	Integer indicating the maximum expected number of subdivisions along any of the coordinates in the grid under construction.
ElemNameLength	Integer variable defining the number of characters in the element names. It may be either 5 or 8. If unequal to 8, ElemNameLength is internally resest to the default (= 5).
FormatType	Character variable indicating the format of the data in the MESH file to be created. It my have one of two values: = 'Old' : This option creates a **MESH** file that conforms to the data format described in Sections 7.2 and 7.3, and is consistent with the TOUGH [*Pruess*, 1987] and TOUGH2 formats [*Pruess*, 1991; *Pruess et al*., 1999]. = 'New' : This option creates a **MESH** file in which the element and connection data are listed using the NAMELIST format facility available in FORTRAN 90/95/2003. Inputs in the next generation of TOUGH+ (currently in development) will make extensive use of NAMELIST-based formats. **NOTE**: *The current version (v1.2) of TOUGH+ and T+H does not accept* NAMELIST-based *formats in the* **MESH** *file. Thus, the option* FormatType = 'Old' *must be used in all* **MeshMaker.f95** *applications*.

7.4.2. Inputs Related to Domain Heterogeneity

These optional inputs provide information that allows the description of heterogeneity within the domain. The assignment of heterogeneity is based on the definition of *regions* using geometrical information that describe the location and extent of these subdomains. Thus, each of the individually defined regions is assigned the properties of a particular medium that will have to be included in the **`ROCKS`** data block in the input file of the subsequent TOUGH+ simulation.

The records in this data block are discussed in detail below:

Record `MESHMAKER.3` (**Optional**)

This record includes only the keyword (character variable) '`Regions`', which is read using a free format. If the domain is homogeneous, then there is no need to provide any of the inputs discussed in Section 7.4.2.

Record `MESHMAKER.3.1` (**Optional** – when the `MESHMAKER.3` record is included)

This record is read using a free format, and includes the single integer variable `Num_HetRegions` (`> 0`) that describes the number of heterogeneous regions (subdomains) that are to be described by the ensuing data records. The minimum value that `Num_HetRegions` can accept is `1`, corresponding to a homogeneous system. An error will occur and the execution will stop if `Num_HetRegions < 1`.

Because of dynamic dimensioning, there is no limit in the number of heterogeneous regions defined by `Num_HetRegions`. However, practical considerations may limit the size of `Num_HetRegions`. Thus, although it is possible to define element-by-element heterogeneity using this approach, this would be a very tedious process. Generally speaking, the most useful application of this facility is in the description of extensive geologic units with distinctly different properties.

Record `MESHMAKER.3.2` (**Optional** – when the `MESHMAKER.3` record is included)

This record is read using a free format, and includes the single character variable `dominant_medium` that provides the name (5 character long) of the dominant (reference) porous medium in the heterogeneous domain. Note that the selection of a medium as `dominant_medium` is be arbitrary, as there are no restrictions on the extent of its spatial distribution for it to be designated as such.

If `Num_HetRegions = 1`, no more data need to be read. This case is equivalent to a homogeneous system, and the entire heterogeneity-related data block (records `MESHMAKER.3` to `MESHMAKER.3.2` may be omitted.

Record `MESHMAKER.3.3.0` (**Optional** – when the `MESHMAKER.3` record is included and `Num_HetRegions > 1`)

This record is read using a free format, and includes the single character variable `H_RegionNname` that provides the name (5 character long) of the porous medium in the region (subdomain) that is about to be defined.

Record `MESHMAKER.3.3.1` (**Optional** – when the `MESHMAKER.3` record is included and `Num_HetRegions > 1`)

The following character variables are read in `MESHMAKER.3.3.1` using a free format:

```
H_RegionCoordinates, H_RegionUnits
```

These parameters are defined as follows:

`H_RegionCoordinates` A character variable indicating the coordinate system used in the geometric definition of the region that is about to be described in the heterogeneous domain. It my have one of two values:

= `'Cartesian'` : This option indicates that the region is defined geometrically in terms of Cartesian coordinates.

= `'Cylindrical'` : This option indicates that the region is defined geometrically in terms of cylindrical coordinates.

`H_RegionUnits` A character variable describing the units of length used in the description of the geometry of the region. The following values are acceptable options:

'mm', 'km', 'km', 'in', or 'ft',

indicating millimeters, meters, kilometers, inches and feet, respectively. Note that **`MeshMaker.f95`** converts all length units into meters (the length unit used in the TOUGH+ simulations) prior to producing the **`MESH`** file.

Record `MESHMAKER.3.3.2` (**Optional** – when the `MESHMAKER.3` record is included and `Num_HetRegions > 1`)

If `H_RegionCoordinates = 'Cartesian'`, the following real variables are read in `MESHMAKER.3.3.2` using a free format:

```
Xmin, Xmax, Ymin, Ymax, Zmin, Zmax
```

These parameters are defined as follows:

`Xmin, Xmax` Real variables indicating the range of the region along the *x*-axis of the Cartesian coordinate system.

`Ymin, Ymax` Real variables indicating the range of the region along the *y*-axis of the Cartesian coordinate system.

`Zmin, Zmax` Real variables indicating the range of the region along the *z*-axis of the Cartesian coordinate system.

If `H_RegionCoordinates = 'Cylindrical'`, the following real variables are read in `MESHMAKER.3.3.2` using a free format:

```
Rmin, Rmax, Zmin, Zmax
```

These parameters are defined as follows:

`Rmin, Rmax` Real variables indicating the range of the region along the *r*-axis of the cylindrical coordinate system.

`Zmin, Zmax` Real variables indicating the range of the region along the *z*-axis of the cylindrical coordinate system.

Repeat records `MESHMAKER.3.3.0`, `MESHMAKER.3.3.1`, `MESHMAKER.3.3.3` for a total of `Num_HetRegions - 1` regions. Note that the region `dominant_medium` does not need geometric definition.

7.4.3. Inputs Related to Description of Boundaries

These optional inputs provide information that describes the outer boundaries of the domain under discretization. The assignment of grid subdomains as boundaries is based on the geometry-based definition of the outer spatial limits of the domain under discretization. Thus, the boundaries are treated as special types of regions (see Section 7.4.2) with the properties of a particular medium that will have to be included in the **`ROCKS`** data block in the input file of the subsequent TOUGH+ simulation. It is possible for a region and one or more boundaries to be described by the same medium in the **`ROCKS`** data block.

The records in this data block are discussed in detail below:

Record `MESHMAKER.4` (**Optional**)

This record includes only the keyword (character variable) '`Boundaries`', which is read using a free format. If the domain is confined by no-flow (Newman-type) boundaries, then there is no need to provide any of the inputs discussed in Section 7.4.3.

Record `MESHMAKER.4.1` (**Optional** – when the `MESHMAKER.4` record is included)

This record is read using a free format, and includes the single integer variable `Num_Boundaries` (`> 0`) that describes the number of boundaries that are to be described.

Record `MESHMAKER.4.2.0` (**Optional** – when the `MESHMAKER.4` record is included and `Num_Boundaries > 1`)

The following character variables are read in `MESHMAKER.4.2` using a free format:

```
BoundID, BoundRegionName
```

These parameters are defined as follows:

`BoundID`	A character variable indicating the type of boundary described at the prescribed location. It my have one of two values: = `'I'` : This option indicates an *inactive* boundary, the conditions and properties of which are time-invariant. = `'V'` : This option indicates a time-variable boundary. The result of this designation is reflected in the **`ELEME`** block, in which `elem_activity` (see Section 7.2) is set to `BoundID` in all the cells corresponding to the boundary defined here. Any other value of the `BoundID` variable causes the program to print an error message and stop execution.
`BoundRegionName`	A character variable `H_RegionNname` that provides the name (5 character long) of the porous medium in the boundary that is about to be defined.

Record `MESHMAKER.4.2.1` (**Optional** – when the `MESHMAKER.4` record is included and `Num_Boundaries > 1`)

The following character variables are read in `MESHMAKER.4.2.1` using a free format:

```
BoundRegionCoordinates, BoundRegionUnits
```

These parameters are defined as follows:

`BoundRegionCoordinates` A character variable indicating the coordinate system used in the geometric definition of the boundary that is about to be described in the heterogeneous domain. It my have one of two values:

= '`Cartesian`' : This option indicates that the region is defined geometrically in terms of Cartesian coordinates.

= '`Cylindrical`' : This option indicates that the region is defined geometrically in terms of cylindrical coordinates.

`BoundRegionUnits` A character variable describing the units of length used in the description of the geometry of the boundary. The following values are acceptable options:

'`mm`', '`km`', '`km`', '`in`', or '`ft`',

indicating millimeters, meters, kilometers, inches and feet, respectively. Note that **`MeshMaker.f95`** converts all length units into meters (the length unit used in the TOUGH+ simulations) prior to producing the **`MESH`** file.

Record `MESHMAKER.4.2.2` (**Optional** – when the `MESHMAKER.4` record is included and `Num_Boundaries > 1`)

If `BoundRegionCoordinates = 'Cartesian'`, the following real variables are read in `MESHMAKER.3.3.2` using a free format:

```
Xmin, Xmax, Ymin, Ymax, Zmin, Zmax
```

These parameters are defined as follows:

`Xmin, Xmax`	Real variables indicating the range of the boundary region along the *x*-axis of the Cartesian coordinate system.
`Ymin, Ymax`	Real variables indicating the range of the boundary region along the *y*-axis of the Cartesian coordinate system.
`Zmin, Zmax`	Real variables indicating the range of the boundary region along the *z*-axis of the Cartesian coordinate system.

If `BoundRegionCoordinates = 'Cylindrical'`, the following real variables are read in `MESHMAKER.3.3.2` using a free format:

```
Rmin, Rmax, Zmin, Zmax
```

These parameters are defined as follows:

`Rmin, Rmax`	Real variables indicating the range of the boundary region along the *r*-axis of the cylindrical coordinate system.
`Zmin, Zmax`	Real variables indicating the range of the boundary region along the *z*-axis of the cylindrical coordinate system.

Repeat records `MESHMAKER.4.2.0`, `MESHMAKER.4.2.1`, `MESHMAKER.4.2.3` for a total of `Num_Boundaries` boudaries.

7.4.4. Inputs for Grid Construction

There are three grid construction options available in **`MeshMaker.f95`**. These options are activated by appropriate keywords. Thus, the keywords '`RZ2D`' or '`RZ2DL`' invoke generation of a one or two-dimensional radially symmetric (*r,z*) mesh; '`XYZ`' initiates generation of a one, two, or three dimensional Cartesian (*x,y,z*) mesh; and '`MINC`' calls a modified version of the GMINC program [*Pruess*, 1983] to sub-partition a primary

porous medium mesh into a secondary mesh for fractured media, using the method of *Multiple INteracting Continua* [*Pruess and Narasimhan*, 1982; 1985].

The meshes generated under keyword 'RZ2D' or 'XYZ' are internally written to file **MESH**. The 'MINC' processing operates on the data in file **MESH**, so that invoking the 'RZ2D' or 'XYZ' options, or assignment of **ELEME** and **CONNE** blocks in the **INPUT** file, must precede the **MeshMaker.f95**/MINC data. We shall now separately describe the preparation of input data for the three grid construction options.

7.4.4.1. Generation of radially symmetric grids (keyword 'RZ2D' or 'RZ2DL'). The keywords 'RZ2D' or 'RZ2DL' invoke generation of a radially symmetric mesh. Values for the radii to which the grid blocks extend can be provided by the user or can be generated internally (see below). Nodal points will be placed halfway between neighboring radial interfaces. When 'RZ2D' is specified, the mesh will be generated by columns; i.e., in the **ELEME** block we will first have the grid blocks at smallest radius for all layers, then the next largest radius for all layers, and so on.

With keyword 'RZ2DL' the mesh will be generated by layers; i.e., in the **ELEME** block we will first have all grid blocks for the first (top) layer from smallest to largest radius, then all grid blocks for the second layer, and so on. Apart from the different ordering of elements, the two meshes for 'RZ2D' and 'RZ2DL' are identical. The reason for providing the two alternatives is as a convenience to users in implementing boundary conditions by way of inactive elements (see Section 6.4 in *Pruess et al*. [1991]). Assignment of inactive elements would be made by using a text editor on the RZ2D-generated **MESH** file, and moving groups of elements towards the end of the **ELEME**

block, past a dummy element with zero volume. 'RZ2D' makes it easy to declare a vertical column inactive, facilitating assignment of boundary conditions in the vertical, such as a gravitationally equilibrated pressure gradient. 'RZ2DL' on the other hand facilitates implementation of areal (top and bottom layer) boundary conditions.

The inputs for cylindrical systems are as follows:

Data Block GRID

The first record in this data block includes only the keyword (character variable) 'RZ2D' or 'RZ2DL'. This keyword is read using a Format (A5).

Record RADII.0

RADII	A keyword that introduces data for defining a set of interfaces (grid block boundaries) in the radial direction. It is read using a Format (A5).

Record RADII.1

Format (I5)
NRAD

NRAD	Number of radii that will be read. At least one radius must be provided, indicating the inner boundary of the mesh.

Record RADII.2, RADII.3, etc.

Format (8E10.4)
RC(i), i = 1,NRAD

RC(i)	The radii defining the element boundaries in ascending order [m].

Record EQUID.0

EQUIDistant	Keyword indicating that the ensuing data describe a set of equal radial increments. This keyword is read using Format (A5).

Record `EQUID.1`

Format (I5, 5X, E10.4)
`NEQU, DR`

`NEQU` The number of desired radial increments.

`DR` The size of the radial increment [m].

NOTE: *At least one radius must have been defined via block* `RADII` *before* `EQUID` *can be invoked*.

Record `LOGAR.0`

`LOGAR``ithmic` This keyword introduces data on radial increments that increase from one to the next by the same factor (i.e., $\Delta r_{n+1} = f \bullet \Delta r_n$).

Record `LOGAR.1`

Format(I5, 5X, 2E10.4)
`NLOG, RLOG, DR`

`NLOG` The number of desired radial increments.

`RLOG` The desired radius r_{max} of the last (largest) of these radii.

`DR` The reference radial increment Δr_0: the first Δr generated will be equal to $f \bullet \Delta r_0$, with f internally determined such that the last increment will bring total radius to `RLOG` = r_{max}. The factor $f < 1$ for decreasing radial increments is permissible. If Δr_0 is set equal to zero, or left blank, the last increment Δr generated before the keyword **`LOGAR`** is invoked will be used as default.

Additional blocks `RADII`, `EQUID`, and `LOGAR` can be specified in arbitrary order.

NOTE: *At least one radius must have been defined before the* `LOGAR` *option can be invoked. If* $\Delta r_0 = 0$*, at least two radii must have been defined.*

Record `LAYER.0`

`LAYER` This keyword introduces information on horizontal layers, and signals closure of `RZ2D` input data. It is read using a Format (A5).

Record `LAYER.1`

Format (I5)
`NLAY`

`NLAY` The number of horizontal layers in the cylindrical grid.

Record `LAYER.2`

Format (8E10.4)
`H(i), i = 1, NLAY`

`H(i)` The thicknesses of the horizontal layers in the cylindrical grid, from top layer downward. By default, zero or blank entries for layer thickness will result in assignment of the last preceding non-zero entry. Assignment of a zero layer thickness, as needed for inactive layers, can be accomplished by specifying a negative value.

NOTE: *The* `LAYER` *data close the* `RZ2D` *or* `RZ2DL` *data block. Note that one blank record must follow to indicate termination of the* `GRID` *data block. Alternatively, the keyword* '`MINC`' *can appear to invoke* `MINC`*-processing for fractured media (see below).*

7.4.4.2. Generation of rectilinear grids (keyword '`XYZ`')

Data Block `GRID`

The first record in this data block includes only the keyword (character variable) '`XYZ`' that is invokes generation of a Cartesian (rectilinear) mesh. This keyword is read using a Format (A5).

Record `XYZ.1`

Format (E10.4)
`DEG`

`DEG` The angle (in degrees) between the *y*-axis and the horizontal. If gravitational acceleration (parameter `gravity` in record `PARAM.2`, see Section 10) is specified positive, -90° < `DEG` < 90° corresponds to grid layers going from top down. Grids can be specified from the bottom layer up by setting `gravity` or `ConxBeta` (Section 7.3) to negative values. The default (`DEG = 0.0e0`) corresponds to horizontal *y*- and vertical *z*-axis. The *x*-axis is always horizontal.

Record `XYZ.2`

Format (A2, 3X, I5, E10.4)

`NTYPE, NO, DEL`

`NTYPE`	A character variable that can assume one of the values '`NX`', '`NY`' or '`NZ`', specifying grid increments in the *x*-, *y*-, or *z*-direction, respectively.
`NO`	The number of grid increments.
`DEL`	The constant grid increment for `NO` grid blocks, if set to a non-zero value.

Record XYZ.3 (**Optional**, `DEL = 0.0e0` or blank only)

Format (8E10.4)

`DEL(i), i = 1, NO`

`DEL(i)`	A set of grid increments in the direction specified by `NTYPE` in record `XYZ.2`. Additional records with formats as `XYZ.2` and `XYZ.3` can be provided, with *x*-, *y*-, and *z*-data in arbitrary order.

Record `XYZ.4` A blank record closes the XYZ data block.

NOTE: *The end of the data block* `GRID` *is also marked by a blank record. Thus, when* `GRID`/`XYZ` *is used, there will be two blank records at the end of the corresponding input data block.*

7.4.4.3. `MINC` processing for fractured media (keyword '`MINC`')

Data Block `GRID`

The first record in this data block includes only the keyword (character variable) '`MINC`' that is invokes post-processing of a primary porous medium mesh from a previously developed file **`MESH`**. This keyword is read using a Format (A5). The input formats in data block `MINC` are identical to those of the GMINC program [Pruess, 1983], with two enhancements: (a) there is an additional facility for specifying global matrix-matrix connections (*dual permeability* option); (b) only active elements (see Section 8.4) will be subjected to `MINC`-processing, the remainder of the **`MESH`** remaining unaltered as porous medium grid blocks. See Section 2.14 for further discussion.

Record MINC.1

Format (2A5, 5X, A5)
PART, TYPE, DUAL

PART — This is the first keyword following the 'MINC' keyword. It will be followed on the same line by parameters TYPE and DUAL with information on the nature of fracture distributions and matrix-matrix connections.

PART — An identifier of the data block with partitioning parameters for secondary mesh.

TYPE — A five-character variable for selecting one of the following six different proximity functions provided in MINC [*Pruess*, 1983].

= 'ONE-D': A set of plane parallel infinite fractures with matrix block thickness between neighboring fractures equal to PAR(1).

= 'TWO-D': Two sets of plane parallel infinite fractures, with arbitrary angle between them. Matrix block thickness is PAR(1) for the first set, and PAR(2) for the second set. If PAR(2) is not specified explicitly, it will be set equal to PAR(1).

= 'THRED': Three sets of plane parallel infinite fractures at right angles, with matrix block dimensions of PAR(1), PAR(2), and PAR(3), respectively. If PAR(2) and/or PAR(3) are not explicitly specified, they will be set equal to PAR(1) and/or PAR(2), respectively.

= 'STANA': Average proximity function for rock loading of Stanford large reservoir model [*Lam et al.,* 1988].

= 'STANB': Proximity function for the five bottom layers of Stanford large reservoir model.

= 'STANT': Proximity function for top layer of Stanford large reservoir model.

DUAL — A five-character word for selecting the treatment of global matrix flow.

= ' ' (Blank – default): The global flow occurs only through the fracture continuum, while rock matrix and fractures interact locally by means of interporosity flow (*double-porosity* model).

= 'MMVER': The global matrix-matrix flow is permitted only in the vertical; otherwise like the double-porosity model; for internal consistency this choice should only be made for flow systems with one or two predominantly vertical fracture sets.

= 'MMALL': The global matrix-matrix flow in all directions; for internal consistency only two continua, representing matrix and fractures, should be specified (dual-permeability model).

NOTE: *A user wishing to employ a different proximity function other than the options provided through the* TYPE *variable in MINC needs to replace the function subprogram* PROX(x) *in* **MeshMaker.f95** *with a routine of the form:*

```
FUNCTION PROX(x)
PROX = (arithmetic expression in x)
RETURN
END
```

It is necessary that PROX(x) *be defined even when x exceeds the maximum possible distance from the fractures, in which case* PROX = 1. *Additionally, when the users supply their own proximity function subprogram, the parameter* TYPE *must be set equal to* 'ONE-D', 'TWO-D', *or* 'THRED', *depending on the dimensionality of the proximity function. This will assure proper definition of innermost nodal distance* [*Pruess,* 1983].

Record PART.l

Format (2I3, A4, 7E10.4)
J, NVOL, WHERE, (PAR(i), i = 1, 7)

J — The total number of multiple interacting continua (J < 36).

NVOL — The total number of explicitly provided volume fractions (NVOL < J). If NVOL < J, the volume fractions with indices NVOL+1, ..., J will be internally generated; all being equal and chosen such as to yield proper normalization to 1.

`WHERE`	Character variable specifying whether the sequentially specified volume fractions begin with the fractures (`WHERE = 'OUT'`) or in the interior of the matrix blocks (`WHERE = 'IN'`).
`PAR(i)`	Real array that stores the parameters describing the fracture spacing (see discussion of previous record).

Record `PART.2.1, PART.2.2, etc.`

Format (8E10.4)
`(VOL(i), i = 1, NVOL)`

`VOL(i)`	The volume fraction (having a value between 0 and 1) of a continuum with index `i` (for `WHERE = 'OUT'`) or index `J+l-i` (for `WHERE = 'IN'`). `NVOL` volume fractions will be read. For `WHERE = 'OUT'`, `i=1` is the fracture continuum, `i=2` is the matrix continuum closest to the fractures, `i=3` is the matrix continuum adjacent to `i=2`, etc. The sum of all volume fractions must not exceed 1.

```
Meshmaker test: Cartesian grid
1000 20  5 'Old'
XYZ
       00.
NX      10    1.0e00
NY       5       2.0
NZ      20       1.0


====> ====> ====> ====>
```

Figure 7.1. An example of a **`MeshMaker.f95`** input file for the creation of a Cartesian 3D grid. Note that no heterogeneous regions or boundaries are defined in this grid.

```
Meshmaker test: Cylindrical grid
6000  6000  5 'Old'
RZ2DL
RADII
    2
    1.0e-5    1.0e-3
EQUID
 4000         2.5e-2
LOGAR
  500         2.5e+3
EQUID
    1         1.0e-3
LAYER
    1
    1.0e00


====> ====> ====> ====>
```

Figure 7.2. An example of a **`MeshMaker.f95`** input file for the creation of a single-layer (1D) cylindrical grid. Note that no heterogeneous regions or boundaries are defined in this grid.

```
Input file for a large 3D cartesian grid       ! Title
500000 100 5  'Old'                    ! MaxNum_Elem, Longest, ElemNameLength, FormatType
Regions
 6                                             ! ==> # of heterogeneous media regions
'HydrL'                                        ! Region #1: Name of dominant medium
'Aquif'                                        ! Region #2: medium name
'cartesian'     'm'                            !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3 -6.3e1 -4.825e1        !    Xmin, Xmax, Ymin, Ymax, Zmin, Zmax
'OverB'                                        ! Region #3: medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3  -3.0e1  0.0e1         !    Rmin, Rmax, Zmin, Zmax
'UndrB'                                        ! Region #4: medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3  -9.4e1 -6.3e1         !    Rmin, Rmax, Zmin, Zmax
'WellA'                                        ! Region #5: medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 5.0e-2 0.0e0 5.0e-2 -5.4e1 -3.0e1        !    Rmin, Rmax, Zmin, Zmax
'WellA'                                        ! Region #6: medium name
'cartesian'   'm'                              !    coordinates, units
4.9995e2 5.0e2 4.9995e2 5.0e2 -5.4e1 -3.0e1    !    Rmin, Rmax, Zmin, Zmax
Boundaries
2                                              ! ==> # of boundaries
'I' 'TopBB'                                    ! Boundary #1: type and medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3 -1.0e-2  0.0e0         !    Rmin, Rmax, Zmin, Zmax
'I' 'BotBB'                                    ! Boundary #2: type and medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3 -9.4e1 -9.3e1          !    Rmin, Rmax, Zmin, Zmax
XYZ
       00.
NX      76
    5.0e-2 2.5000e-1 2.8599e-1 3.2716e-1 3.7426e-1 4.2813e-1 4.8976e-1 5.6027e-1
 6.4092e-1 7.3319e-1 8.3873e-1 9.5947e-1 1.0976e+0 1.2556e+0 1.4363e+0 1.6431e+0
 1.8797e+0 2.1502e+0 2.4598e+0 2.8139e+0 3.2190e+0 3.6823e+0 4.2124e+0 4.8188e+0
 5.5125e+0 6.3061e+0 7.2139e+0 8.2524e+0 9.4404e+0 1.0799e+1 1.2354e+1 1.4132e+1
 1.6167e+1 1.8494e+1 2.1157e+1 2.4202e+1 2.7686e+1 3.1672e+1 3.1672e+1 2.7686e+1
 2.4202e+1 2.1157e+1 1.8494e+1 1.6167e+1 1.4132e+1 1.2354e+1 1.0799e+1 9.4404e+0
 8.2524e+0 7.2139e+0 6.3061e+0 5.5125e+0 4.8188e+0 4.2124e+0 3.6823e+0 3.2190e+0
 2.8139e+0 2.4598e+0 2.1502e+0 1.8797e+0 1.6431e+0 1.4363e+0 1.2556e+0 1.0976e+0
 9.5947e-1 8.3873e-1 7.3319e-1 6.4092e-1 5.6027e-1 4.8976e-1 4.2813e-1 3.7426e-1
 3.2716e-1 2.8599e-1 2.5000e-1    5.0e-2
NY      76
    5.0e-2 2.5000e-1 2.8599e-1 3.2716e-1 3.7426e-1 4.2813e-1 4.8976e-1 5.6027e-1
 6.4092e-1 7.3319e-1 8.3873e-1 9.5947e-1 1.0976e+0 1.2556e+0 1.4363e+0 1.6431e+0
 1.8797e+0 2.1502e+0 2.4598e+0 2.8139e+0 3.2190e+0 3.6823e+0 4.2124e+0 4.8188e+0
 5.5125e+0 6.3061e+0 7.2139e+0 8.2524e+0 9.4404e+0 1.0799e+1 1.2354e+1 1.4132e+1
 1.6167e+1 1.8494e+1 2.1157e+1 2.4202e+1 2.7686e+1 3.1672e+1 3.1672e+1 2.7686e+1
 2.4202e+1 2.1157e+1 1.8494e+1 1.6167e+1 1.4132e+1 1.2354e+1 1.0799e+1 9.4404e+0
 8.2524e+0 7.2139e+0 6.3061e+0 5.5125e+0 4.8188e+0 4.2124e+0 3.6823e+0 3.2190e+0
 2.8139e+0 2.4598e+0 2.1502e+0 1.8797e+0 1.6431e+0 1.4363e+0 1.2556e+0 1.0976e+0
 9.5947e-1 8.3873e-1 7.3319e-1 6.4092e-1 5.6027e-1 4.8976e-1 4.2813e-1 3.7426e-1
 3.2716e-1 2.8599e-1 2.5000e-1    5.0e-2
NZ      77
    1.0e-3    7.00e0    5.00e0    4.0e+0    3.2e+0    2.5e+0    2.0e00    1.6e00
   1.25e00    1.0e00    8.0e-1    6.5e-1    5.0e-1    5.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.5e-1    4.5e-1    4.5e-1    4.5e-1    4.5e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    5.0e-1    5.0e-1    6.5e-1    8.0e-1    1.0e00
    1.0e00   1.25e00    1.6e00    2.0e00    2.5e00    3.2e00    3.0e00    4.00e0
    4.0e00    5.0e00    6.0e00    8.0e00    1.0e-3


====> ====> ====> ====>
```

Figure 7.3. An example of a **`MeshMaker.f95`** input file for the creation of a large Cartesian 3D grid with heterogeneous regions and defined boundaries.

```
Input file for a large cylindrical grid  ! Title
15000 30000 5 'Old'           ! MaxNum_Elem, Longest, ElemNameLength, FormatType
Regions                       ! Keyword denoting heterogeneous subdomains
  6                           ! => Num_HetRegions ( = number of heterogeneous regions)
'HydrL'                       !    Region #1: dominant_medium
'Aquif'                       !    Region #2: H_RegionName
'cylindrical'   'm'           !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.5e3 -6.3e1 -4.825e1   !       Rmin, Rmax, Zmin, Zmax
'OverB'                       !    Region #3: H_RegionName
'cylindrical'   'm'           !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.5e3 -3.0e1  0.0e1     !       Rmin, Rmax, Zmin, Zmax
'UndrB'                       !    Region #4: H_RegionName
'cylindrical'   'm'           !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.5e3 -9.4e1 -6.3e1     !       Rmin, Rmax, Zmin, Zmax
'Casng'                       !    Region #5: H_RegionName
'cylindrical'   'm '          !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.0e-1 -4.6e1 -3.0e1    !       Rmin, Rmax, Zmin, Zmax
'Perfo'                       !    Region #5: H_RegionName
'cylindrical'   'm '          !       H_RegionCoordinates, H_RegionUnits
0.0e0  1.0e-1 -5.225e1 -4.6e1 !       Rmin, Rmax, Zmin, Zmax
Boundaries                    ! Keyword denoting boundaries
2                             ! ==> Num_Boundaries
'I' 'TopBB'                   !    Boundary #1: BoundID, BoundRegionName
'cylindrical'   'm'           !       BoundRegionCoordinates, BoundRegionUnits
0.0e0 1.5e3 -1.0e-2  0.0e0    !       Rmin, Rmax, Zmin, Zmax
'I' 'BotBB'                   !    Boundary #2: BoundID, BoundRegionName
'cylindrical'   'm'           !       BoundRegionCoordinates, BoundRegionUnits
0.0e0 1.5e3 -9.4e1 -9.3e1     !       Rmin, Rmax, Zmin, Zmax
RZ2DL
RADII
    2
     0.001   0.10795
EQUID
    1         2.0e-2
LOGAR
  199         1.0e+3
LAYER
  68
    1.0e-3    6.50e0    5.00e0    5.0e+0    3.0e+0    3.0e+0    2.0e00    2.0e00
    1.0e00    1.0e00    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.5e-1    5.5e-1    5.5e-1    5.5e-1
    5.5e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    1.0e00
    1.0e00    2.0e00    2.0e00    3.0e+0    3.0e+0    4.0e+0    4.0e+0    5.00e0
    5.0e+0    6.00e0    6.0e+0    1.0e-3


====> ====> ====> ====>
```

Figure 7.4. An example of a **`MeshMaker.f95`** input file for the creation of a large cylindrical 2D grid with multiple layers, heterogeneous regions and defined boundaries.

8.0. Initial Conditions and Boundary Conditions

In this section the data blocks that allow for domain-specific initial conditions (block **`INDOM`**), element-specific initial conditions (**`INCON`**), extended capabilities for specifying initial conditions (**`EXT-INCON`**), including features for assigning initial conditions to ranges of elements (defined by a variety of methods). Following that the procedure is described for implementing initial conditions and boundary conditions in TOUGH+.

8.1. Data Block `INCON`

This block introduces element-specific initial conditions.

<u>Record `INCON.1` (5-character element names)</u>

Format (A5, 2I5, E15.8, 2x, A3, 36x, 3(E15.8))
`ElName5C, NSEQ, NADD, porosity,`

```
StateIndex, (perm(i), i=1,3)
```

`ElName5C`	The 5-character name of the element that is being initialized.
`NSEQ`	The number of additional elements with the same initial conditions.
`NADD`	The increment between the code numbers of two successive elements with identical initial conditions.
`porosity`	The porosity of the element that is being initialized. If `porosity` is zero or blank, the element porosity will be taken as specified in block **`ROCKS`** or **`MEDIA`**. This feature is necessary for assignment of element-specific properties for the description of highly or statistically heterogeneous domains.
`StateIndex`	State identifier (see Section 3.1, **Tables 3.1** and **3.2**): the initial conditions corresponding to this identifier are applied uniformly over the element.
`perm(i)`	The intrinsic (absolute) permeabilities of the element that is being initialized along the three directions described by the `ConxKi` variable (see Section 7.3). If all `perm(i), i=1,…,3` are zero or blank, the element permeabilities will be taken as specified in block **`ROCKS`** or **`MEDIA`**. This feature is necessary for assignment of element-specific properties for the description of highly or statistically heterogeneous domains.

<u>Record `INCON.2`</u> specifies primary variables.

Format (6E20.13)
`X(i), i = 1,NumCom+1`

The primary variables defining the state of the element specified in record `INCON.1`. **`INCON`** specifications will supersede default conditions specified in `PARAM.4` (see Section 10), and domain-specific conditions that may have been specified in data block **`INDOM`**. See Section 3.1 (**Tables 3.1** and **3.2**) for a detailed description of the potential sets of primary variables.

<u>Record `INCON.3`</u>

There are three ways to close the **INCON** data block.

If the simulation is not a continuation run, then a blank line or a record with '<<<' typed in the first three columns closes the **INCON** data block.

In continuation runs from a previous TOUGH [*Pruess*, 1987] or TOUGH2 [*Pruess*, 1991; *Pruess et al*., 1999] simulation, the **INCON** data block may terminate on a record with '+++' typed in the first three columns, followed by data that describe timing and time stepping information.

In continuation runs from a previous TOUGH+, the **INCON** data block may terminate on a record with '::: ' typed in the first three columns. This is followed by a data describing (a) the origin of time and the simulated time at the conclusion of the preceding TOUGH+ simulation, (b) the number of timesteps and Newtonian iterations, and (c) accumulated quantities, e.g., total amounts of fluids produced/injected during the previous run (see **Figure 8.1**). These data are recorded automatically at the end of the **SAVE** file (see Section 8.4) upon completion of the previous TOUGH+ run using a NAMELIST format. This is a very powerful format that allows maximum clarity and flexibility, accepting free formats, arbitrary ordering of variables, insertions of comments anywhere in the input fields, and providing the option of ignoring any of the NAMELIST parameters by not assigning a value to it. For more information, the reader is directed to a textbook on FORTRAN 95/2003.

<u>Record INCON.1 (8-character element names)</u>

Format (A8, 7X, E15.8, 2x, A3, 36x, 3(E15.8))

```
ElName8C, NSEQ, NADD, porosity,
StateIndex, (perm(i), i=1,3)
```

ElName8C	The 8-character name of the element that is being initialized.

All other variables remain as in the case of five-character element names.

```
.
.
.
.
FJ000           0.00000000E+00  Aqu: P,     X_m_A, X_i_A, T
 3.2500495961160E+07 0.0000000000000E+00 3.0000000000000E-02 9.3974540000000E+00
FJ106           0.00000000E+00  Aqu: P,     X_m_A, X_i_A, T
 3.3467136230360E+07 2.9244292873310E-03 3.0000000000690E-02 1.2627840000000E+01
:::
 &Data_For_Continuation_Run
       timesteps_to_this_point     =       6818,
       NR_iterations_to_this_point =      71226,
       number_of_detected_media    =          8,
       origin_of_time              =  0.00000000000000E+00,
       time_to_this_point          =  4.69736464257198E+07,
       accumulated_quantities      =  7.35454091207164E+04,  1.08644220169487E+05,
                                      1.03974248845536E+05,  6.43009774059305E+04,
                                      0.00000000000000E+00,  0.00000000000000E+00,
                                   /
<<<
```

Figure 8.1. An example of the NAMELIST-described termination data printed at the end of the **SAVE** file from a TOUGH+ simulation. These data can be read as part of the INCON data block, or of the INCON external file. The names of the variables defined in the NAMELIST are self-explanatory. For reference, this figure lists the conditions in the last two elements (FJ000 and FJ106) of the grid in the TOUGH+ simulation.

8.2. Data Block INDOM

This block introduces domain-specific initial conditions. These will supersede default initial conditions specified in PARAM.4 (see Section 10), and can be overwritten by element-specific initial conditions in data block **INCON** or data block **EXT-I**NCON. The option **START** is needed to use **INDOM** conditions.

Record INDOM.1

Format (A5, 2X, A3)
Rk_name, StateIndex

Rk_name	The name of a medium (corresponding to a system subdomain), as specified in data block **ROCKS**.
StateIndex	The state identifier describing the conditions applying to the Rk_name medium/subdomain.

Record INDOM.2

Format (6E20.13)
`X(i), i = 1,NumCom+1`

`X(i)` are the primary variables assigned to all grid blocks in the domain specified in record `INDOM.1`. See Section 3.1 (**Tables 3.1** and **3.2**) for description of the potential sets of primary variables.

Record `INDOM.3`

A blank record closes the **`INDOM`** data block.

Repeat records `INDOM.1` and `INDOM.2` for as many domains as desired. The ordering is arbitrary and need not be the same as in block **`ROCKS`**.

8.3. Data Block `EXT-INCON`

This block introduces extended capabilities for specifying initial conditions of subdomains (i.e., groups of element). The user has several options to provide element information (e.g., an element name or number list, location, element sequence, columnar structure, etc.) that defines a subdomain to be initialized with the specified conditions. The first entry is the total number of initialization entities entered within the block. Following that, each of the various input entities may be entered in an arbitrary order.

Record `EXT-INCON.0`

This record includes the single integer variable `Total_input_num` that describes the total number of input entities (representing initial conditions in particular subdomains) that will be entered. This is read using a free format.

Record `EXT-INCON.1`

This record includes the single character variable `INTYPE` that is read using a free format, and which provides a keyword defining the type of data describing the subdomain to be initialized. `INTYPE` can assume one of the following values:

'GEOMETRY', 'LIST', 'SEQUENCE', or 'COLUMN'. Thus, INTYPE determines what kind of variables/data will be read in record in EXT-INCON.2.

8.3.1. Data Block **GEOMETRY**

The data in this block are read when INTYPE = 'GEOMETRY', and define a set of elements in a subdomain that is bounded within prescribed minimum and maximum coordinates. All elements within this range will be assigned the initial conditions entered in the record EXT-INCON.3.

Record EXT-INCON.2.0

For a Cartesian grid (coordinate_system = 'Cylindrical', see Section 5), the following real variables are read in EXT-INCON.2 using a free format:

```
Xmin, Xmax, Ymin, Ymax, Zmin, Zmax
```

These parameters are defined as follows:

Xmin, Xmax	Real variables indicating the range of the subdomain to-be-initialized along the *x*-axis of the Cartesian coordinate system.
Ymin, Ymax	Real variables indicating the range of the subdomain to-be-initialized along the *y*-axis of the Cartesian coordinate system.
Zmin, Zmax	Real variables indicating the range of the subdomain to-be-initialized along the *z*-axis of the Cartesian coordinate system.

For a cylindrical grid (coordinate_system = 'Cartesian', see Section 5), the following real variables are read in EXT-INCON.2 using a free format:

```
Rmin, Rmax, Zmin, Zmax
```

These parameters are defined as follows:

`Rmin, Rmax`	Real variables indicating the range of the subdomain to-be-initialized along the *r*-axis of the cylindrical coordinate system.
`Zmin, Zmax`	Real variables indicating the range of the subdomain to-be-initialized along the *z*-axis of the cylindrical coordinate system.

8.3.2. Data Block **SEQUENCE**

For `INTYPE = 'SEQUENCE'`

Record `EXT-INCON.2.0`

Format (*), i.e., free format
`SequICFirstElemNum, SequICLastElemNum, SequICStride`

`SequICFirstElemNum`	Integer describing the global number of the first element in the sequence.
`SequICLastElemNum`	Integer describing the global number of the last element in the sequence.
`SequICStride`	Integer describing the stride in the numbering sequence

Thus, a sequence of elements is defined by the beginning and ending element number, as well as by the stride (number of elements to skip between two successive elements in the subdomain defined by the sequence). For example, a sequence with `SequICFirstElemNum = 10`, `SequICLastElemNum = 20`, and `SequICStride = 2` results in the a group (subdomain) of elements with the following global element numbers: 10, 12, 14, 16, 18, 20. If `SequICLastElemNum < SequICFirstElemNum` or if `SequICStride < 0`, an error message is printed and the simulation is aborted.

All elements within this sequence will be assigned the initial conditions entered in the record `EXT-INCON.3`.

8.3.3. Data Block **LIST**

For `INTYPE = 'LIST'`

Record `EXT-INCON.2.0`

Format (*), i.e., free format
`ListICLength, N_per_row`

`ListICLength`	Integer denoting the length of the list (i.e., the total number of element numbers in the list) that is about to be read.
`N_per_row`	Integer defining the number of entries (= element numbers) per row in the list that that is about to be read. The last row may have fewer than `N_per_row` entries.

Record `EXT-INCON.2.1`

Format (*), i.e., free format
`ElemNum(i), i=1, ListICLength`

NOTE: *The number of entries per row of the* `ElemNum(i)` *input data is* `N_per_row`

The element numbers that are defined through their participation in a list are read in rows (= records), each (except possibly the last one) containing `N_per_row` entries. The last row/record may have fewer than `N_per_row` entries. For example, if `ListICLength = 8` and `N_per_row = 3`, then the element numbers would be listed as follows:

```
ElemNum(1),ElemNum(2),ElemNum(3),   ! 1st record
ElemNum(4),ElemNum(5),ElemNum(6),   ! 2nd record
ElemNum(7),ElemNum(8)               ! 3rd record
```

All elements in this list will be assigned the initial conditions entered in the following record (`EXT-INCON.3`).

8.3.4. Remaining Data Blocks in **EXT-INCON**

For INTYPE = 'GEOMETRY', 'SEQUENCE', or 'LIST'

Record EXT-INCON.3

Format (*), i.e., free format
StateIndex, X0(i),i=1,NumCom+1

StateIndex	A character variable indicating the state index (see Section 3, **Tables 3.1** and **3.2**) corresponding to the initial conditions that are to be assigned to the subdomain defined by INTYPE.
X0(i)	A real array that includes the primary variables (corresponding to StateInd) that describe the conditions to which the respective subdomain is initialized. See Section 3.1 for a thorough description of the potential sets of primary variables.

Repeat records EXT-INCON.3, EXT-INCON.2 and EXT-INCON.3 for a total of Total_input_num subdomain-based initializations.

8.3.5. Data Block **COLUMN**

This data block is somewhat different from the previous ones, and is very useful in applying initial conditions in problems that involve initialization after achieving gravity equilibration in a single column of the domain, which then serves as the reference column. Thus, when INTYPE = 'COLUMN', the following records and data are read:

Record EXT-INCON.2.0

This record includes the single integer variable ColmICSize that is read using a free format, and which describes the column length, i.e., the number of elements in the column.

Record `EXT-INCON.2.1`

Format (*)

```
ColmICFirstElemNum, ColmICNumElemInCol,
ColmICstride
```

`ColmICFirstElemNum`	An integer variable indicating the global element number of the first element in the columnar structure to be initialized.
`ColmICNumElemInCol`	An integer variable describing the total number of elements to be initialized using the columnar structure. Thus, the total number of columns to be initialized using the data provided in this block is: `ColmICNumElemInCol/ColmICSize`
`ColmICstride`	An integer variable describing the stride in the numbering sequence. This number is the difference between the global numbers of two elements that two successive locations in the same column.

For example, if `ColmICSize=10`, `ColmICFirstElemNum=1`, `ColmICNumElemInCol=80`, and `ColmICstride=1`, then initialization using the `COLUMN` data block will assign the initial conditions (obtained from the 10 elements of the reference column) to columns composed of elements with the following global numbers: Column #1, elements 1 to 10; Column #1, elements 11 to 20; … Column #8, elements 71 to 80. In this case, elements 1, 11, 21,…,71 have the same initial conditions (equal to those of the first entry in the reference column). Similarly, elements 2, 12, 22, …, 72 have all the same initial conditions (equal to those of the second entry in the reference column).

Conversely, if `ColmICstride=1`, then elements 1 to 8 have all the same initial conditions (equal to those of the first entry in the reference column), 2 to 16 have all the same initial conditions (equal to those of the second entry in the reference column), etc.

Record `EXT-INCON.3.x, x=1,…,ColmICSize`

Format (*)

```
StateIndex, (X0(i),i=1,NumCom+1)
```

A total of `ColmICSize` records are read, each providing the state index and primary variables of the elements of the reference column. The variables in these records are as previously defined (see Section 8.3.4).

8.4. Implementing Initial Conditions

Flow systems are initialized by assigning a complete set of primary thermodynamic variables to all grid blocks into which the flow domain is discretized. Various options are available in a hierarchical system, as follows. During the initialization of a TOUGH+ run, all grid blocks are first assigned to default thermodynamic conditions specified in data block **PARAM**. The defaults can be overwritten for selected reservoir domains by assigning domain-specific conditions in data blocks **INDOM** or **EXT_INCON**. These in turn may be superseded by thermodynamic conditions assigned to individual grid blocks in data block **INCON**. A disk file **INCON** written to the same specifications as data block **INCON** may also be used.

The possible sets of primary variables are discussed in Section 3.1 (**Tables 3.1** and **3.2**), with the actual primary variables depending on the fluid/solid phase composition. During phase change primary variables will be automatically switched from one set to another. In multiphase flow systems, therefore, different grid blocks will in general have different sets of primary variables, and must be initialized accordingly.

For many applications, special initial conditions are needed, such as gravity-capillary equilibrium, or steady state corresponding to certain mass and heat flows. This can be realized by performing a series of TOUGH+ runs, in which thermodynamic conditions obtained in one run, and written to disk file **SAVE**, are used as initial conditions in a subsequent continuation run. For example, in a hydrate accumulation simulation, a first run may be made to obtain hydrostatic pressure conditions. These may subsequently be used as boundary conditions in a second run segment to simulate undisturbed natural

state conditions with through-flow of mass and heat. This could be followed by a third run segment with fluid production and injection.

Restarting of a TOUGH+ run is accomplished by renaming the file **SAVE** generated in a previous run as file **INCON** for initialization. Usually additional (often minor) adjustments will be made for a restart. For example, different specifications for the number of time steps and desired printout times may be made. Some editing of the **MESH** file may be needed to make certain grid blocks inactive, so that previously calculated pressures can serve as boundary conditions (see below). In a continuation run, simulation time and time step counters may be continuously incremented, or they may be reset to zero. For example, the latter option will be used when simulating production and injection operations following preparation of a natural initial state, which may correspond to a large simulation time.

As far as the internal workings of the code is concerned, there is no difference between a fresh start of a simulation and a restart. The only feature that makes a simulation a continuation run is that the **INCON** data were generated by a previous TOUGH2 or TOUGH+ run, rather than having them explicitly provided by the user.

The file **SAVE** originating from TOUGH2 simulations [*Pruess et al*., 1991] always ends with a data record with '+++' in the first three columns, followed by one record with restart information (time step and iteration counters, simulation time). The file **SAVE** originating from a TOUGH+ simulation terminates with continuation data written using the NAMELIST format shown in **Figure 8.1**. In either case, to reset all counters and continuation data to zero when using **SAVE** as file **INCON** for another TOUGH+ run,

users can simply replace all records below the conditions of the last element with a single blank record.

8.5. Implementing Boundary Conditions

Boundary conditions can be of two basic types. Dirichlet conditions prescribe thermodynamic conditions, such as pressure, temperature, etc. on the boundary, while Neumann conditions prescribe fluxes of mass or heat crossing boundary surfaces. A special case of Neumann boundary conditions is *no flux*, which is the default in the integral finite difference framework when no flow connections are specified across the boundary. More general flux conditions are prescribed by introducing sinks or sources of appropriate strength into the elements adjacent to the boundary.

In TOUGH2 [*Pruess et al*., 1999], Dirichlet conditions could be implemented by assigning very large volumes (e.g., $V = 10^{50}$ m^3, as described by the `elem_vol` variable in Section 7.2) to grid blocks adjacent to the boundary, so that their thermodynamic conditions do not change at all from fluid or heat exchange with finite-size blocks in the flow domain. In addition, a small value (such as 10^{-9} m) should be specified for the nodal distance (`ConxD1` or `ConxD2`, see Section 7.3) of such blocks, so that boundary conditions are in fact maintained in close proximity to the surface where they are desired, and not at some distance from it. It is possible to specify a nodal distance that is outright zero; however, this may interfere with options for the computation of interface mobilities that are intended for modeling fracture-matrix interactions. Therefore, assigning zero nodal distances (`ConxD1` or `ConxD2`) should be used with caution.

For time-independent Dirichlet boundary conditions, TOUGH2 offered an alternative implementation, which provided savings in computational work along with added user conveniences in running simulation problems. This was accomplished by defining *active* and *inactive* elements. By convention, elements encountered in data block **ELEME** (or in geometry files **MESH** or **MINC**) were taken to be active until the first element entry with a zero or negative volume was encountered. The first element with volume `elem_vol` ≤ `0.0e0`, and all subsequent elements, were by convention taken to be inactive. The easiest way to declare selected grid blocks as inactive was to use a text editor to move them to the end of the **ELEME** data block, and then insert a dummy grid block of zero volume in front of them.

TOUGH+ maintains these two older options available in TOUGH2, but provides an additional one that is much simpler as it does not require any editing of the element volume or physically moving any portions of the element list. In TOUGH+, elements can be designated as inactive if the parameter `element_activity` in record `ELEME.1` (see Section 7.2) is set to `'I'` (when its conditions are time-invariant) or `'V'` (when its conditions and properties vary over time). Then, these elements, as well as all elements designated as inactive by the older TOUGH2 options, are treated as inactive.

For the inactive elements no mass or energy balance equations are set up, their primary thermodynamic variables are not included in the list of unknowns, and their thermodynamic conditions remain unchanged during the course of the simulation. Inactive elements can appear in flow connections and initial condition specifications like all other elements. The computational overhead of inactive elements is small because they do not increase the number of equations to be solved in a flow problem.

9.0. Sources and Sinks

9.1. Data Block GENER

This block introduces sinks and/or sources to the system

Record GENER.1 (for 5-character element names)

Format (A5, A5, 4I5, 5X,A4, A1, 3E10.4 ,A4, 6x, 3(E10.4))

```
ElName5C, SS_name, NSEQ, NADD, NADS, LTAB,
SS_Type, ITAB, GX, EX, HX,
WellResponse, PresLimits,
RateStepChange, RateLimit
```

ElName5C	The code name of the 5-character element containing the sink/source.
SS_name	The name of the sink/source. The first three characters are arbitrary, the last two characters must be numbers.
NSEQ	The number of additional sinks/sources with the same injection/production rate (not applicable for SS_Type = 'DELV').

`NADD` The increment between the code numbers of two successive elements with identical sink/source.

`NADS` The increment between the code numbers of two successive sinks/sources.

`LTAB` The number of points in table of generation rate versus time. Set 0 or 1 for constant generation rate. For wells on deliverability, `LTAB` denotes the number of open layers, to be specified only for the bottommost layer.

`SS_Type` The type of source or sink. This variable specifies different options for fluid or heat production and injection. For example, different fluid components may be injected, the nature of which depends on the EOS module being used. Different options for considering wellbore flow effects may also be specified. The following options are available:

- = `'HEAT'`: Introduces a heat sink/source. This option is to be used for injection only.
- = `'COM1'`: Indicates mass component #1 (usually water). This option is to be used for injection only.
- = `'WATE'`: Indicates water injection.
- = `'COM2'`: Indicates mass component #2. This option is to be used for injection only.
- = `'COM3'`: Indicates mass component #3. This option is to be used for injection only.
- = `'COMn'`: Indicates mass component #n. This option is to be used for injection only.
- = `'MASS'`: Specified a mass production rate, i.e., the mass rate of all the fluids withdrawn from a system.
- = `' DELV'`: Well on deliverability, i.e., production occurs against specified bottomhole pressure. If well is completed in more than one layer, bottommost layer must be specified first, with number of layers given in

LTAB. Subsequent layers must be given sequentially for a total number of LTAB layers.

F--- or f--- well on deliverability against specified wellhead pressure. By convention, when the first letter of a type specification is F or f, TOUGH+ will perform flowing wellbore pressure corrections using tabular data of flowing bottomhole pressure vs. flow rate and flowing enthalpy. The tabular data used for flowing wellbore correction must be generated by means of a wellbore simulator ahead of a TOUGH+ run according to certain format specifications (see **Table 17**, section 7.3 in *Pruess et al.*, 1991). The data must be provided by means of a disk file, whose name consists of the four characters of the SS_Type specification, and the one character of the following ITAB parameter. For example, to use wellbore pressure data in a disk file called **f120a**, specify SS_Type as 'f120', and specify ITAB as 'a'. Different wellbore tables, representing e.g. wells with different diameter, feed zone depth, and flowing wellhead pressure, may be used simultaneously in a TOUGH+ simulation. Also, several wells completed in different grid blocks may reference the same wellbore table.

The capability for flowing wellbore pressure correction is presently only available for wells with a single feed zone.

ITAB — Unless left blank, a table of specific enthalpies will be read (LTAB > 1 only).

GX — The constant generation rate. GX is positive for injection and negative for production. GX describes a mass rate [kg/sec] for generation types COM1, COM2, COM3, etc., and MASS; it describes an energy rate [W] for a HEAT sink/source. For wells on deliverability, GX is the productivity index PI [m^3] – see Equation (9.2).

EX — The fixed specific enthalpy [J/kg] of the fluid for mass injection (GX>0). For wells on deliverability against fixed bottomhole pressure, EX is the

bottomhole pressure P_{wb} [Pa] at the center of the topmost producing layer in which the well is open.

`HG` — The thickness of layer [m]. This is to be used only in cases of wells on deliverability with specified bottomhole pressure.

`WellResponse` — A character variable that describes how the source/sink is to be treated if pressure limits (as described by the variable `PresLimits`) at the corresponding cell are violated. The following options are available:

- `= 'STOP'`: The simulation is halted.
- `= 'ZERO'`: The source/sink rate `GX` is reset to zero, and the simulation continues.
- `= 'ADJU'`: The source/sink rate `GX` is adjusted, and the simulation continues. Note that a simulation involving the `WellResponse = 'ADJU'` option may involve several successive `GX` adjustments.

`PresLimits` — A real variable describing the pressure limit [Pa] that acts as a trigger for `WellResponse` to be enacted. For injection, pressure in the element containing the source has to exceed `PresLimits` for changes to be implemented. For production, pressure in the element containing the sink has to fall below `PresLimits` for changes to be implemented.

`RateStepChange` — A real variable that is used only when `WellResponse = 'ADJU'`. By convention, `RateStepChange` is always a positive number. It represents the absolute value of the fraction by which `GX` is to decrease when the `PresLimits` criterion is violated. For obvious reasons, $0 \leq$ `RateStepChange` ≤ 1.

`RateLimit` — A real variable describing the minimum rate limit, described as the lowest fraction of the original `GX` to which the rate is allowed to decline. The simulation stops when `RateLimit` is reached.

<u>Record `GENER.1` (for 8-character element names)</u>

Format (A8, A5, 12X, I5, 5X,A4, A1, 3E10.4 ,A4, 6x, 3(E10.4))

```
ElName8C, SS_name, LTAB,
SS_Type, ITAB, GX, EX, HX,
WellResponse, PresLimits,
RateStepChange, RateLimit
```

`ElName8C` The code name of the 8-character element containing the sink/source.

All other variables and parameters are as in the case of the 5-character elements.

Record `GENER.l.l` (**Optional**, `LTAB > l` only)

Format (4E14.7)
`Fl(k), k=l, LTAB`

`F1(k)` Generation times in the table of the time variable source/sink data.

Record `GENER.1.2` (**Optional**, `LTAB > l` only)

Format (4E14.7)
`F2(k), k=1, LTAB`

`F2(k)` Generation rates in the table of the time variable source/sink data.

Record `GENER.1.3` (**Optional**, `LTAB > l` and `ITAB` non-blank only)

Format (4E14.7)
`F3(k), k=1, LTAB`

`F3(k)` Specific enthalpy of produced or injected fluid in the table of the time variable source/sink data.

Repeat records `GENER.1`, `GENER.1.1`, `GENER.1.2`, and `GENER.1.3` for the desired number of sinks/sources.

Record `GENER.2`

A blank record closes the **`GENER`** data block. Alternatively, generation information may terminate on a record with '+++' typed in the first three columns, followed by data describing the numbers of the elements where the sources/sinks are located.

9.2. Discussion on sinks and sources

Sinks and sources are introduced through data block **GENER** in the input file. Several options are available for specifying the production ($q < 0$) or injection ($q > 0$) of fluids and heat. Any of the mass components may be injected in an element at a constant rate, or at time-dependent rates that may be prescribed through user-defined tables. The user has the option of specifying the specific enthalpy of the injected fluid as either a constant or time dependent value. Heat sources/sinks (with no mass injection) may be either constant or time-dependent.

Fluid production from an element may be handled by prescribing a constant or time-dependent mass rate. In this case, the phase composition of the produced fluid may be determined by the relative phase mobilities in the source element. Alternatively, the produced phase composition may be specified to be the same as the phase composition in the producing element. In either case, the mass fractions of the components in the produced phases are determined by the corresponding component mass fractions in the producing element. Different options are available for interpolating time-dependent rates from user-supplied tabular data; these may be selected through parameter `MOP(12)` – see Section 10.

9.2.1. Deliverability Model

Production wells may operate on deliverability against a prescribed flowing bottomhole pressure P_{wb} with a productivity index PI [*Coats*, 1977]. With this option, the mass production rate of phase β from a grid block with phase pressure $P_\beta > P_{wb}$ is

$$q_\beta = \frac{k_{r\beta}}{\mu_\beta} \rho_\beta \cdot PI \cdot (P_\beta - P_{wb}) \tag{9.1}$$

For steady radial flow the productivity index is given by [*Coats*, 1977; *Thomas*, 1982]

$$(PI)_l = \frac{2\pi (k\Delta z_l)_l}{\ln\left(\frac{r_e}{r_w}\right) + s - \frac{1}{2}} \tag{9.2}$$

Here, Δz_l denotes the layer thickness, $(k\ \Delta z)_l$ is the permeability-thickness product in layer l, r_e is the grid block radius, r_w is the well radius, and s is the skin factor. If the well produces from a grid block that does not have cylindrical shape, an approximate PI can be computed by using an effective radius

$$r_e = \sqrt{\frac{A}{\pi}} \tag{9.3}$$

where A is the grid block area; e.g., $A = \Delta x\ \Delta y$ for an areal Cartesian grid. More accurate expressions for specific well patterns and grid block shapes have been given in the literature [e.g., *Peaceman*, 1977, 1982; *Coats and Ramesh*, 1982].

The rate of production for mass component κ is

$$\hat{q}^\kappa = \sum_\beta X_\beta^\kappa q_\beta \tag{9.4}$$

For wells which are screened in more than one layer (element), the flowing wellbore pressure P_{wb} can be corrected to approximately account for gravity effects according to

the depth-dependent flowing density in the wellbore. Assume that the open interval extends from layer l = 1 at the bottom to l = L at the top. The flowing wellbore pressure in layer l, $P_{wb,l}$, is obtained from the wellbore pressure in layer l + 1 immediately above it by means of the following recursion formula

$$P_{wb,l} = P_{wb,l+1} + \frac{g}{2}\left(\rho_l^f \, \Delta z_l + \rho_{l+1}^f \, \Delta z_{l+1}\right) \tag{9.5}$$

Here, g is acceleration of gravity, and ρ_l^f is the flowing density in the tubing opposite layer l. Flowing densities are computed using a procedure given by *Coats* [private communication with *K. Pruess*, 1982]. If wellbore pressure were zero, we would obtain the following volumetric production rate of phase β from layer l.

$$r_{l,\beta} = \left(\frac{k_{r\beta}}{\mu_\beta}\right)_l (PI)_l \, P_{l,\beta} \tag{9.6}$$

The total volumetric flow rate of phase β opposite layer l is, for zero wellbore pressure

$$r_{l,\beta}^T = \sum_{m=1}^{l} r_{m,\beta} \tag{9.7}$$

From this we obtain an approximate expression for flowing density opposite layer l, which can be used in Equation (9.5).

$$\rho_l^f = \sum_\beta \rho_{l,\beta} \, r_{l,\beta}^T \Big/ \sum_\beta r_{l,\beta}^T \tag{9.8}$$

During fluid production or injection, the rate of heat removal/injection is determined by

$$\hat{q}^h = \sum_\beta q_\beta \, h_\beta \tag{9.9}$$

where h_β is the specific enthalpy of phase β.

9.2.2. Coupled Wellbore Flow

Geothermal production wells typically operate at (nearly) constant wellhead pressures. As flow rate and flowing enthalpy change with time, wellbore pressure gradients and flowing bottomhole pressures will also change. From a conceptual point of view, the most straightforward way to describe production from geothermal wells is to set up and solve equations for flow in the reservoir and flow in the wellbore in a fully coupled manner. This approach was taken by *Hadgu et al*. [1995], who coupled the reservoir simulator TOUGH [*Pruess*, 1987] with the wellbore simulator WFSA [*Hadgu and Freeston*, 1990].

As discussed by *Murray and Gunn* [1993], an alternative approach may be preferable in which the wellbore and reservoir simulations are performed separately. This can be accomplished by running a wellbore flow simulator prior to the reservoir simulation for a range of flow rates q and flowing enthalpies h, in order to generate a table of flowing bottomhole pressures P_{wb}.

$$P_{wb} = P_{wb}\left(q,h;P_{wh},z,r_w\right) \tag{9.10}$$

In addition to the functional dependence on q and h, flowing bottomhole pressure is dependent on a number of well parameters. These include wellhead pressure P_{wh}, feed zone depth z, wellbore radius r_w, friction factors, and possibly others. By interpolating on these tabular data, Equation (9.10) can be directly inserted into the well source term, Equation (9.1). Reservoir flow equations that include a quasi-steady approximation to wellbore flow can then be solved with little added computational expense compared to the case where no wellbore flow effects are considered. Advantages of representing wellbore flow effects through tabular data include increased robustness and

computational efficiency. It also makes it possible to use different wellbore simulators and two-phase flow correlations without any programming changes in the reservoir simulation code.

We have incorporated a tabular interpolation scheme for dynamic changes of flowing bottomhole pressure into TOUGH+. Flowing enthalpy at the well feed is known from phase mobilities and enthalpies calculated by the reservoir simulator. The unknown well flow rate and flowing bottomhole pressure are obtained by Newton-Raphson iteration on

$$R(q) \equiv q - \left(\sum \frac{k_{r\beta}}{\mu_\beta} \rho_\beta \right) \cdot PI \cdot \left[P - P_{wb}(q,h) \right] = 0 \qquad (9.11)$$

The iterative solution of Equation (9.11) was embedded in the outer (Newtonian) iteration performed by TOUGH+ on the coupled mass and heat balance equations. Additional computational work in comparison to conventional simulations with constant downhole pressure is insignificant.

Coupled wellbore flow is invoked in a TOUGH+ simulation by specifying a **GENER**-type starting with the letters `'F'` or `'f'`. For example, `'f725d'` is a valid **GENER**-type that will associate the host element with a production well. If this is specified, then the user must supply a disk file named `f725d` with appropriately formatted data that relate wellhead to flowing bottomhole pressures. Several wells may use the same wellbore pressure table, and multiple data sets for flowing wellbore pressures may be used, e.g., for wells with different diameter, feed zone depth, and wellhead pressure. The well table(s) must be generated by wellbore simulation prior to a TOUGH+ run as ASCII data of 80 characters per record, as described in the example given in Table 17 of *Pruess et al.* [1991].

The first record is an arbitrary title. The second record holds the number of flow rate and flowing enthalpy data (table points), `NG` and `NH`, respectively, in Format 2I5; in Table 17 of *Pruess et al.* [1991] we have `NG=11`, `NH=9`. This is followed by `NG` flow rate data in format 8E10.4, and `NH` enthalpy data also in format 8E10.4. After this come `NG` sets of `NH` flowing bottomhole pressure data in format 8E10.4. The data in Table 17 of *Pruess et al.* [1991] were generated with the HOLA wellbore simulator [*Aunzo et al*., 1991] for a 0.2 m (≈ 8 inch) inside diameter well of 1,000 m feed zone depth with 7 bars wellhead pressure. Formation temperature for the conductive heat loss calculation in HOLA was assumed to increase linearly from 25 ˚C at the land surface to 275.5 ˚C at 750 m depth. Flow rates cover the range from 0.5 to 90.5 kg/s, and flowing enthalpies cover the range from 1,000 to 1,400 kJ/kg. A data record with very large bottomhole pressures of 55.55 MPa was added by hand for a very large hypothetical rate of 1,000 kg/s. This was done to avoid rates going out of table range during the Newton-Raphson iteration in a TOUGH2 flow simulation, thus resulting in more robust time stepping behavior (see Section 9.5 and Figure 10 in *Pruess et al.* [1991]).

The coupled wellbore flow capability as coded in TOUGH+ is limited to wells with a single feed zone and can only handle wellbore pressure effects from changing flow rates and enthalpies. Effects from changing fluid composition, as e.g. variable non-condensible gas content, are not modeled at present.

10. Computational Parameters

In this section the data blocks (**PARAM** and **SOLVR**) that specify computational parameters are described and followed by a discussion on linear equation solvers.

10.1. Data Block PARAM

This block describes an assortment of computation parameters.

Record PARAM.1

Format (2I2, 3I4, 24I1, 3E10.4, 2I5)

```
Max_NumNRIterations, OutputOption,
Max_NumTimeSteps, iCPU_MaxTime,
PRINT_frequency, (MOP(i),i=1,24),
BaseDiffusionCoef, DiffusionExpon,
DiffusionStrength,
SAVE_frequency, TimeSeries_frequency
```

`Max_NumNRIterations` Integer specifying the maximum number of Newtonian iterations per time step (default is 8)

`OutputOption` Integer defining an option that controls the amount of printout in the standard TOUGH+ output (the default is 1). The following options are available:

`= 0,1:` Print a selection of the most important variables.

`= 2:` In addition, print the mass and heat fluxes and the flow velocities.

`= 3:` In addition, print the primary variables and their changes.

If the above values for `OutputOption` are increased by 10, printout will occur after each Newton-Raphson iteration (not just after convergence).

`Max_NumTimeSteps` An integer defining the maximum number of time steps allowed during the simulation.

NOTE: *If* `Max_NumTimeSteps < 0`, *then the maximum number of timesteps becomes* `1000*ABS(Max_NumTimeSteps)`

`iCPU_MaxTime` An integer describing the maximum duration, in CPU seconds, of the simulation (default is infinite).

`PRINT_frequency` A integer describing the printout frequency. This, printout will occur for every multiple of `PRINT_frequency` steps (the default is `1`).

NOTE: *If* `PRINT_frequency < 0`, *then the printout frequency becomes* `1000*ABS(PRINT_frequency)`

`MOP(i), i=1,24` An integer array that allows choice of various computational options. These are described in detail below, and are documented in the printed standard output from a TOUGH+ run.

`MOP(1)` If `MOP(1)` $\neq$ `0`, a short printout for non-convergent iterations will be generated at the end of each Newton-Raphson iteration.

`MOP(2)` to `MOP(6)` If $\neq$ `0`, these options generate additional printout in various subroutines at the end of each Newton-

raphson iteration. This feature is not needed in normal applications, but can be convenient in the development of new capabilities as they can be used to identify coding errors. The amount of printout increases with the value of `MOP(i)`. The user is encouraged to consult the source code listings for details. Below we list the subroutines corresponding to `MOP(2)` to `MOP(6)`:

`MOP(2)` **`Simulation_Cycle`** (executive subroutine that advances time and controls the flow of data during the simulation)

`MOP(3)` **`JACOBIAN_SetUp`** (routine computing the flow and accumulation terms in the mass and energy balance equations).

`MOP(4)` **`SourceSink_Equation_Terms`** (subroutine determining the contribution of sinks/sources to the mass and energy balance equations).

`MOP(5)` **`Equation_Of_State`** (routine describing the equation of state of the system under investigation and computing all thermophysical properties).

`MOP(6)` **`Solve_Jacobian_Matrix_Equation`** (routine that solves the linear equations of the Jacobian matrix).

`MOP(7)` If `MOP(7)` $\neq$ `0`, a printout of the input data is provided in the standard output file.

`MOP(8)` It determines how relative permeability and capillary pressure are estimated in the presence of solid phases (see discussion of Section 2.12). The following options are available:

- `= 0:` Based on the OPM model; capillary pressure scaling based on EPM #1.
- `= 1:` Based on EPM #1 model; capillary pressure scaling based on EPM #1.
- `= 2:` Based on EPM #2 model; capillary pressure scaling based on EPM #2.

= 3: Based on EPM #1 model, no capillary pressure scaling.

= 4: Based on EPM #2 model, no capillary pressure scaling.

= 9: Based on OPM model, no capillary pressure scaling.

MOP(9) It determines the composition of produced fluid with the MASS option - see discussion on the data block **GENER** in Section 9. The relative amounts of phases are determined as follows:

= 0: according to relative mobilities in the source element.

= 1: the produced source fluid has the same phase composition as the producing element.

MOP(10) It controls the selection of the interpolation formula for the composite heat conductivity as a function of the various phase saturations. The following options are available

= 0: $k_\Theta = k_{dry} + (\sqrt{S_A} + \sqrt{S_H})(k_{wet} - k_{dry}) + \phi S_I \lambda_I$

= 1: $k_\Theta = k_{dry} + (S_A + S_H)(k_{wet} - k_{dry}) + \phi S_I \lambda_I$

= 2: $k_\Theta = k_{dry} + \phi\left(S_A\lambda_A + S_H\lambda_H + S_I\lambda_I\right)$ (Based on the linear model of *Bejan* [1984], gas contribution ignored)

= 3: $k_\Theta = k_{dry} + \phi\left(S_A\lambda_A + S_H\lambda_H + S_I\lambda_I + S_G\lambda_G\right)$ (Based on the linear model of *Bejan* [1984], including gas contribution)

Options MOP(10)=0 and MOP(10)=1 are based on extensions of an earlier model of *Somerton et al*. [2003; 2004] based on the analysis of *Moridis et al*. [2005] of the thermal properties of hydrates from laboratory studies [*Kneafsey et al*., 2005]. It is not known under what conditions (if any) the linear model of *Bejan* [1984] (invoked for MOP(10)= 2 and MOP(10)= 3) is applicable, but it is

included for completeness. The option `MOP(10)= 3` is discouraged because of (a) doubts about the validity of the *Bejan* [1984] linear model, (b) the very demanding computations for the estimation of the gas thermal conductivity from the real gas property package in **T+H** is demanding, and (c) the small overall contribution to the composite thermal conductivity.

`MOP(11)` It provides alternative options for the evaluation of mobility and permeability at interfaces. These are:

- `= 0:` The mobilities are upstream weighted according to the `W_upstream` factor (see discussion in `PARAM.3`), and the permeability is upstream-weighted.
- `= 1:` The mobilities are averaged between adjacent elements, and the permeability is upstream-weighted.
- `= 2:` The mobilities are upstream weighted, and the permeability is harmonic-weighted.
- `= 3:` The mobilities are averaged between adjacent elements, and the permeability is harmonic-weighted.
- `= 4:` The mobility and permeability are both harmonic weighted.

For multiphase flow simulations in which the upstream element is not known *a priori*, `MOP(11) = 0` and `MOP(11) = 0` are the recommended options. The user is strongly cautioned against using other options.

`MOP(12)` It determines the interpolation procedure of the tabular data (involving times, flow rates and enthalpies, see Section 8) in time-dependent sources and sinks. The following options are available:

- `= 0:` triple linear interpolation; tabular data are used to obtain interpolated rates and enthalpies for the beginning and end of the

time step; the average of these values is then used.

= 1: step function option; rates and enthalpies are taken as averages of the table values corresponding to the beginning and end of the time step.

= 2: rigorous step rate capability for time dependent generation data.

A set of times t_i and generation rates q_i provided in data block **GENER** is interpreted to mean that sink/source rates are piecewise constant and change in discontinuous fashion at table points. Specifically, generation is assumed to occur at constant rate q_i during the time interval $[t_i, t_{i+1})$, and changes to q_{i+1} at t_{i+1}. The actual rate used during a time step that ends at time t, with $t_i \le t \le t_{i+1}$, is automatically adjusted in such a way that total cumulative exchanged mass at time t

$$Q(t) = \int_0^t q\,dt = \sum_{j=1}^{i-1} q_j (t_{j+1} - t_j) + q_i (t - t_i)$$

is rigorously conserved. If tabular data for enthalpies are also provided, an analogous adjustment is made to fluid enthalpy to preserve $\int q_h\,dt$.

MOP(13) Option used in processes involving mechanical dispersion.

MOP(14) It specifies the handling of gas solubility in liquid phases according to one of the following options:

= 0: The gas solubility is computed using Henry's constants (T-independent).

= 1: The gas solubility is computed using appropriate equations of Henry's dissolution parameters (T-dependent).

> 1: The gas solubility is computed using fugacities (T- and P-dependent)

For low *P* and *T* variations, it is possible to use the `MOP(14)=0` option, which leads to the use of a *P*- and *T*-invariant Henry's constants for the estimation of solubility. Because of the pressure and temperature regime in hydrate simulations, this option is unavailable in **T+H**. When `MOP(14)=1`, solubility is computed from fast parametric equations that describe the effect of temperature. For `MOP(14)>1`, gas solubility is computed from fugacities and activity coefficients. While this approach is accurate over the entire region of pressure and temperature regimes handled by **T+H**, it is computationally intensive. In the case of the relatively insoluble CH_4, the incremental improvement over the predictions when `MOP(14)=1` is marginal, and cannot justify the significantly longer execution times. Thus, this option is deactivated in the current version.

`MOP(15)` A flag indicating whether conductive heat exchange with impermeable confining layers (see Section 7.4) is to be considered.

- `= 0:` The heat exchange is not considered.
- `= 1:` The heat exchange is activated (for grid blocks that have a non-zero heat transfer area; see data block **`ELEME`** in Section 7.2).

`MOP(16)` It provides automatic time step control. Time step size will be doubled if convergence occurs within `ITER` ≤ `MOP(16)` Newton-Raphson iterations.

It is recommended to set `MOP(16)` in the range of `2 - 5`. The default value is `4`.

`MOP(17)` It specifies the handling of binary gas diffusivities according to one of the following options:

- `= 0:` The binary gas diffusivity is computed from the method of *Fuller et al.* [1969].
- `= 7:` The basic estimate of the binary gas diffusivity is computed from the method of *Fuller et al.* [1969], and is then adjusted for

high pressures using the method of *Riazi and Whitson* [1993].

`MOP(18)` A flag determining the method for estimating interface density.

`= 0:` Perform upstream weighting for interface density.

`> 0:` Use the average interface density between the two grid blocks. However, when one of the two phase saturations is zero, upstream weighting is to be performed.

`MOP(19)` This is the parameter that controls the simulation output. When `MOP(19)<8`, a standard ASCII file output (as described in general terms by *Pruess et al.* [1999]) is produced. Depending on the value of the `OutputOption` parameter in block `PARAM.1`, this output can includes pressure, temperature and saturation distribution of the various phases, concentrations, thermophysical properties, and primary and secondary variables.

When `MOP(19)=8`, two additional files containing the most important properties are also printed in a format that conforms to the requirements of the TecPlot package [*TecPlot*, 2003], and is suitable for most other plotting and graphing packages. The names of these two files are `Plot_Data_Coord` (containing the Cartesian coordinates of the grid elements) `Plot_Data_Elem` (which stores the element-specific properties and parameters for plotting and graphing). Depending on the value of the `OutputOption` parameter, it also possible to obtain the third file `Plot_Data_Conx` that stores connection-specific properties and parameters for plotting and graphing. For `MOP(19)=9`, the plotting files and a truncated standard output file are produced (listing only mass balances at the prescribed printout times).

`MOP(20)` Flag determining whether the validity of the initial conditions is to be checked. The options are:

`< 9:` The initial conditions are checked to ensure physically meaningful and non-contradictory state indices and the corresponding primary variable values (default).

`= 9:` No checking of initial conditions is performed.

This option may be useful in continuation runs involving large grids, in which case checking of the initial conditions (as provided by the **SAVE** file) is both time consuming (in terms of computation time) and generally unnecessary. In general, the user is discouraged from bypassing the checking process.

`MOP(21)` A computational parameter to selects the linear equation solver (see Section 10.2) from among the following options:

`= 0:` defaults to `MOP(21) = 3`.

`= 1:` `LUBAND`, banded direct solver using LU decomposition.

`= 2:` `DSLUBC`, bi-conjugate gradient solver with preconditioner.

`= 3:` `DSLUCS` (default), Lanczos-type preconditioned bi-conjugate gradient solver with preconditioner.

`= 4:` `DSLUGM`, generalized minimum residual preconditioned conjugate gradient solver with preconditioner.

`= 5:` `DLUSTB`, stabilized bi-conjugate gradient solver with preconditioner.

All conjugate gradient solvers use incomplete LU-factorization as a default preconditioner. Other preconditioners may be chosen by means of the data block **SOLVR** (see Section 10.2).

`MOP(22)` A flag indicating whether an additional output file will be created to list the location and magnitude of the largest pressure in the domain.

= 9: A time series of the maximum pressure in the domain and its location is printed in the external file **MaxP_Time_Series**.

< 9: No max$\{P\}$ output is provided (default).

`MOP(23)` Used by some equations of state. Information on its usage is provided in the User's Manual of the corresponding equations of state.

`MOP(24)` It determines handling of multiphase diffusive fluxes at interfaces according to one of the following options:

= 0: Harmonic weighting of fully coupled effective multiphase diffusivity.

= 1: Separate harmonic weighting of the gas and liquid phase diffusivities.

`BaseDiffusionCoef` The base gas diffusion coefficient [m^2]

`DiffusionExpon` Parameter (exponent) describing the temperature dependence of gas phase diffusion coefficient – see Equation (6.4).

`DiffusionStrength` Parameter (optional) describing the effective strength of enhanced vapor diffusion; if set to a non-zero value, it will replace the parameter group $\phi\tau_0\tau_\beta$ for vapor diffusion – see Equation (2.46) and Section 6.4.1.

`SAVE_frequency` Frequency of writing and saving the **SAVE** file. This feature avoids data loss if the simulation is interrupted. A value between 100 and 500 is recommended. When `SAVE_frequency = 0`, the **SAVE** file is written and stored only once at the conclusion of the simulation.

`TimeSeries_frequency` Frequency of writing and saving the data in the time-series output files (files **SS_Time_Series**, **Elem_Time_Series**, **Conx_Time_Series**, and the various **RegionXX_Time_Series** files). This feature avoids the creation of very large time series files in long simulations that involve

many thousands of time steps. Note that mass-balance related calculations related to some of the parameters included in the time series output files are conducted at the conclusion of each time step, but the results are printed only when specified by this parameter. The default value of `TimeSeries_frequency=1`.

Record `PARAM.2`

Format (4E10.4, A5, 5X, 3E10.4)

```
TimeOrigin, SimulationTimeEnd,
InitialTimeStep, MaxTimeStep, TrackElemName,
gravity, Dt_reducer, scale
```

`TimeOrigin`	A real variable indicating the origin of time (starting time) in the simulation [sec]. The default is `TimeOrigin=0.0e0`.
`SimulationTimeEnd`	A real variable indicating the time [sec] at which simulation should stop. The default is infinite.
`InitialTimeStep`	A real variable specifying the initial time step size [sec]. If `InitialTimeStep<0`, then the program proceeds to read `NumDts = INT(ABS(InitialTimeStep))` records with time step information.
`MaxTimeStep`	A real variable defining the upper limit for time step size [sec]. The default is infinite.
`TrackElemName`	A character variable providing the name of an element, the behavior of which is to be tracked over time by printing a short printout of the evolution of its key conditions and properties after each time step.
`gravity`	A real variable specifying the magnitude [m/sec^2] of the gravitational acceleration vector. Blank or zero gives "no gravity" calculation.
`Dt_reducer`	A real variable defining the factor by which time step is reduced in case of convergence failure or other problems. The default value is `4`.

`scale` The scale factor (a real variable) by which the size of the mesh is adjusted (default = 1.0).

Record `PARAM.2.1, 2.2`, etc.

Format (8E10.4)
`(UserTimeStep(i), i=1, NumDts)`

`UserTimeStep(i)` is the length [sec] of time step `i`.

This set of records is optional, and is read only when `InitialTimeStep<0`, in which case the integer `NumDts = INT(ABS(InitialTimeStep))`. Up to 13 records can be read, each containing 8 time step data. If `Max_NumTimeSteps` > `NumDts`, the simulation will continue with time steps equal to the last non-zero `UserTimeStep(i)` encountered. When automatic time step control is chosen (`MOP(16)>0`), time steps following the last `UserTimeStep(i)` input will increase according to the convergence rate of the Newton-Raphson iteration. Automatic time step reduction will occur if the maximum number of Newton-Raphson iterations is exceeded (parameter `Max_NumNRIterations`, record `PARAM.1`)

Record `PARAM.3`

Format (7E10.4, 2X, A3, 3E10.4)

```
rel_convergence_crit, abs_convergence_crit,
U_p, W_upstream, W_NRIteration,
derivative_increment, W_implicitness,
DefaultStateIndex, P_overshoot, T_overshoot,
S_overshoot
```

`rel_convergence_crit` Convergence criterion for relative error (real variable, parameter ε_1, see Equation (3.8), default = 10^{-5}).

`abs_convergence_crit` Convergence criterion for absolute error (real variable, parameter ε_2, see discussion of Equation (3.9), default = 1).

`U_p` Not used in TOUGH+, maintained only to ensure compatibility with older TOUGH2 input files.

`W_upstream` The upstream weighting factor (real variable) for computing mobilities and enthalpies at interfaces.

The default `W_upstream` = 1.0 is strongly recommended for multi-phase flows. ($0 \leq$ `W_upstream` ≤ 1).

`W_NRIteration` A weighting factor (real variable) determining the level of updating of the solutions based on the results of the Newton/Raphson iteration. The default `W_NRIteration` = 1.0 is recommended. ($0 <$ `W_NRIteration` ≤ 1).

`derivative_increment` The increment factor (a real variable) for numerically computing derivatives. The default value is `derivative_increment` = $10^{-m/2}$, where m, evaluated internally, is the number of significant digits of the floating point processor used. For 64-bit arithmetic, `derivative_increment` $\approx 10^{-8}$.

`W_implicitness` A weighting factor (a real variable) describing the level of implicitness in the solutions. The default `W_implicitness` = 1.0 is recommended. ($0 <$ `W_implicitness` ≤ 1).

`DefaultStateIndex` The default state identifier (a character variable) of the general initial conditions that apply uniformly over the entire if not amended by the data blocks related to the description of initial conditions in subdomains (see Section 8).

`P_overshoot` A real variable specifying the level of *overshoot* (defined as a fraction) allowed in the computation of pressure if P is used as a criterion for triggering phase and state changes. When `P_overshoot = 0.0e0` (default), the P-triggered phase and state changes are at their most accurate and sensitive (*hair-trigger*).

This variable is introduced to alleviate potential problems caused by narrow oscillations about phase equilibrium lines that are possible under certain conditions. In most cases, a value of $10^{-6} \leq$ `P_overshoot` $\leq 10^{-4}$ is sufficient if a hair trigger causes problems. If `P_overshoot` < 0 or `P_overshoot` $\geq 5.0 \times 10^{-2}$, it is rest internally to its default value (=0).

`T_overshoot`	A real variable specifying the level of *overshoot* (defined as a fraction) allowed in the computation of temperature if *T* is used as a criterion for triggering phase and state changes. The definitions, defaults, limits and application are entirely analogous to those of `P_overshoot`.
`S_overshoot`	A real variable specifying the level of *overshoot* (defined as a fraction) allowed in the computation of saturations if thearese used criteria for triggering phase and state changes. The definitions, defaults, limits and application are entirely analogous to those of `P_overshoot`.

Record `PARAM.4`

This record introduces a set of primary variables that are used as default initial conditions for all grid blocks that are not assigned by means of data blocks **`INDOM`**, **`INCON`** or **`EXT-INCON`**. Option **`START`** is necessary to use default **`INCON`**. The format and date read here are:

Format (6E20.13)

```
default_initial_cond(i), i=1,NumCom+1
```

As is self evident, the real variables `default_initial_cond(i)` describe the initial conditions of the state defined by `StateIndex`, as defined by the corresponding primary variables. When more than six primary variables are needed, more than one line (record) must be provided. See Section 3.1 for description of potential sets of state indices and primary variables.

10.2. Modification of Computational Parameters During the Course of a TOUGH+ Simulation

It is possible to modify the computational parameters described in Section 10.1 in the course of a TOUGH+ simulation without having to interrupt the execution. This feature is particularly useful in simulations involving large grids and a large number of timesteps (`Max_NumTimeSteps`) when the user observe solution convergence and time advancement that can be improved by varying some of these computational parameters.

The process is controlled by the parameter `SAVE_frequency` (see Section 10.1, record `PARAM.1`) that determines the frequency of updating the **`SAVE`** file. At the time of updating the **`SAVE`** file, TOUGH+ also interrogates the directory of execution for the presence of a file called **`Parameter_Update_File`**. If no such file exists, there is no updating of the computational parameters.

If a file **`Parameter_Update_File`** exists, then it is opened and the following data are read:

Record `UPDATE.1`

This record includes the single character variable `UpdateHeader` that is read using a free format. The TOUGH+ computational parameters are updated only when `UpdateHeader = 'Update_Simulation_Parameters'`. Otherwise, the **`Parameter_Update_File`** is closed and the simulation continues without any parameter updating.

Record `UPDATE.2`

This record includes a set of real variables (computational parameters) that are to be updated. The data in this record are read using a `NAMELIST` format, and may occupy one or more lines (a choice left to the user). As already discussed (Section 8.1), `NAMELIST`-based formats are a feature of FORTRAN 95/2003 and provide unique power and flexibility, allowing (a) assignment of updated values to any subset of the parameters included in the `NAMELIST` definition, (b) arbitrary order, (c) free formats of individual parameter values, (d) inclusion of

comments, etc. Future versions of TOUGH+ will involve NAMELIST-based formats to read most input data.

The namelist in this record is named Real_Parameters_To_Update, and includes the following real parameters:

(1) SimulationTimeEnd
(2) MaxTimeStep
(3) rel_convergence_crit
(4) abs_convergence_crit
(5) P_overshoot
(6) T_overshoot
(7) S_overshoot

The corresponding TOUGH+ computational parameters will be updated if values are provided for any of these computational parameters. The structure of the Real_Parameters_To_Update namelist (and its use as an input format) is best illustrated in the example of **Figure 10.1**.

Record UPDATE.3

This record includes a set of integer computational parameters that are to be updated. The data in this record are read using a NAMELIST format. This namelist is named Integer_Parameters_To_Update, and includes the following integer parameters:

(1) Max_NumTimeSteps
(2) Max_NumNRIterations
(3) MOP_16
(4) SAVE_frequency
(5) TimeSeries_frequency

The corresponding TOUGH+ integer parameters will be updated if values are provided for any of these computational parameters. It is possible to stop a simulation by providing a Max_NumTimeSteps value that is smaller than its current value in the code. Then, the simulation will be halted upon reading the smaller Max_NumTimeSteps value while at the same time preserving the data in the **SAVE** file, which would be lost if the execution is interrupted.

The structure of the Integer_Parameters_To_Update namelist (and its use as an input format) is best illustrated in the example of **Figure 10.1**.

Upon reading the contents of the **Parameter_Update_File** and updating the computational parameters, TOUGH+ (a) prints a prominent message in the standard output file that provides all the new parameter values, and (b) replaces the UpdateHeader = 'Update_Simulation_Parameters' value with

the value `UpdateHeader = '==> Completed Update # n'`, where `n` is the number of the update. Because `UpdateHeader` has no longer the value that will cause TOUGH+ to read the subsequent data, this substitution prevents multiple readings of the same **`Parameter_Update_File`** while keeping track of the number of updates and preserving the evolution of the updated parameters in the input file. Note that several updates are possible in the course of a long simulation. To accomplish this, the data for the next update are simply added to the top of the **`Parameter_Update_File`** file without erasing the updating history up to this point. The process is clearly illustrated in the example of **Figure 10.2**.

```
'Update_Simulation_Parameters'      ! UpdateHeader
    &Real_Parameters_To_Update    ! Namelist #1
        SimulationTimeEnd    = 5.0d6  ,
        MaxTimeStep          = 3600.  ,
        rel_convergence_crit = 1.5d-5 ,
        P_overshoot          = 1.0e-4 ,
        S_overshoot          = 1.0e-5
        /                      ! Not updated: abs_convergence_crit, T_overshoot
    &Integer_Parameters_To_Update   ! Namelist #2
        Max_NumTimeSteps    =  500   ,
        Max_NumNRIterations =  10000 ,
        MOP_16              =  4     ,
        SAVE_frequency      =  100   ,
        TimeSeries_frequency = 5
        /
```

Figure 10.1. An example of a **`Parameter_Update_File`** for parameter updating in the course of a TOUGH+ simulation. Within the namelists (**`Real_Parameters_To_Update`** and **`Integer_Parameters_To_Update`**), parameters can be entered in any order, data are read using any kind of appropriate format, only the needed parameters are included, and comments can be added.

```
Update_Simulation_Parameters
    &Real_Parameters_To_Update
        SimulationTimeEnd    = 3.0d7
        /
    &Integer_Parameters_To_Update
        Max_NumTimeSteps    =  2000
        /
==> Completed Update #  3             ! 3rd Update
    &Real_Parameters_To_Update
        SimulationTimeEnd    = 2.0d7  ,
        MaxTimeStep          = 8.64e4 ,
        rel_convergence_crit = 2.5d-5 ,
        /
    &Integer_Parameters_To_Update
        Max_NumTimeSteps    =  1500  ,
        Max_NumNRIterations =  20000
        /
==> Completed Update #  2             ! 2nd Update
    &Real_Parameters_To_Update
        SimulationTimeEnd    = 8.0d6  ,
        MaxTimeStep          = 7200.  ,
        rel_convergence_crit = 2.0d-5 ,
        P_overshoot          = 1.0e-5
        /
    &Integer_Parameters_To_Update
        Max_NumTimeSteps    =  1000  ,
        MOP_16              =  5     ,
        SAVE_frequency      =  200   ,
        TimeSeries_frequency = 5
        /
==> Completed Update #  1             ! 1st Update
    &Real_Parameters_To_Update
        SimulationTimeEnd    = 5.0d6  ,
        MaxTimeStep          = 3600.  ,
        rel_convergence_crit = 1.5d-5 ,
        P_overshoot          = 1.0e-4 ,
        S_overshoot          = 1.0e-5
        /
    &Integer_Parameters_To_Update
        Max_NumTimeSteps    =  500   ,
        Max_NumNRIterations =  10000 ,
        MOP_16              =  4     ,
        SAVE_frequency      =  100   ,
        TimeSeries_frequency = 5
        /
```

Figure 10.2. An example of a **`Parameter_Update_File`** indicating three completed parameter updates, in addition to another one (at the top of the file) that has not yet been executed.

10.3. Data Block SOLVR

This (optional) block specifies parameters used by linear equation solvers.

Record `SOLVR.1`

Format(I1,2X,A2,3X,A2,2E10.4)

```
MatrixSolver, Z_preprocessing,
O_preprocessing, Max_CGIterationRatio,
CG_convergence_crit
```

`MatrixSolver` This integer variable selects the linear equation solver from among the following options:

- `= 1:` LUBAND
- `= 2:` DSLUBC
- `= 3:` DSLUCS
- `= 4:` DSLUGM
- `= 5:` DLUSTB

`Z_preprocessing` A character variable that determines the type of Z-preconditioning [*Moridis and Pruess*, 1998]. Regardless of user specifications, Z-preprocessing will only be performed when iterative solvers are used ($2 \leq$ `MatrixSolver` ≤ 5), and if there are zeros on the main diagonal of the Jacobian matrix. The following options are available:

- `='Z0'`: No Z-preprocessing; default for `NumEqu=1` and `MatrixSolver=1`
- `='Z1'`: Replace zeros on the main diagonal by a small constant (1.e-25; default for `NumEqu>1` and for `1<MatrixSolver≤5`)
- `='Z2'`: Make linear combinations of equations for each grid block to produce non-zero main diagonal entries
- `='Z3'`: Normalize equations, followed by `Z2`
- `='Z4'`: Same as in `O_preprocessing='O4'`

`O_preprocessing` A character variable that determines the type of

O-preconditioning [*Moridis and Pruess*, 1998].

`='O0'`: No O-preprocessing; default for `NumEqu=1` and `MatrixSolver=1`

`='O1'`: Elimination of lower half of the main-diagonal submatrix with center pivoting

`='O2'`: `O1` + Elimination of upper half of the main-diagonal submatrix with center pivoting

`='O3'`: `O2` + Normalization; results in unit main-diagonal submatrices

`='O4'`: Pre-processing which results in unit main-diagonal submatrices without center pivoting

`Max_CGIterationRatio` An integer variable that specifies the maximum number of CG iterations as a fraction of the total number of equations (`0` < `Max_CGIterationRatio` ≤ `1`; default is `Max_CGIterationRatio = 0.1`)

`CG_convergence_crit` A real variable that specifies the convergence criterion for the CG iterations (`1.0e-12`≤`CG_convergence_crit`≤`1.0e-6`; the default `CG_convergence_crit=1.0e-6`)

The solver `DLUSTB` implements the BiCGSTAB(m) algorithm [*Sleijpen and Fokkema*, 1993], an extension of the BiCGSTAB algorithm of *van der Vorst* (1992). `DLUSTB` provides improved convergence behavior when iterations are started close to the solution, i.e., near steady state. The preconditoning algorithms can cope with difficult problems in which many of the Jacobian matrix elements on the main diagonal are zero. An example was given in *Pruess et al.* [1991] in "*two-waters*" problems in which typically 2/3 of the elements in the main diagonal are zero. Our tests show that this type of problem can be solved by means of Z2 or Z3 preconditioning [*Moridis and Pruess*, 1998].

10.4. Discussion on Linear Equation Solvers

The computational work to be performed in the course of a TOUGH+ simulation includes evaluation of thermophysical properties for all grid blocks, assembly of the vector of residuals and the Jacobian matrix, and solution of the linear equation system for each Newton-Raphson iteration step. Except for small problems with just a few grid blocks, the most computationally intensive of these different tasks is the solution of the linear equation system. TOUGH+ offers a choice of direct or iterative methods for linear equation solution; technical details of the methods and their performance can be found in *Moridis and Pruess* [1998].

The most reliable linear equation solvers are based on direct methods, while the performance of iterative techniques tends to be problem-specific and lacks the predictability of direct solvers. The robustness of direct solvers comes at the expense of large storage requirements and execution times that typically increase with problem size N (= number of equations solved) proportional to N^3. In contrast, iterative solvers have much lower memory requirements, and their computational work will increase much less rapidly with problem size, approximately proportional to N^{ω}, with $\omega \approx 1.4 - 1.6$ [*Moridis and Pruess*, 1995]. For large problems and especially 3-D problems with several thousand grid blocks or more, iterative conjugate gradient (CG) type solvers are therefore the method of choice.

The default linear equation solution technique in TOUGH+ uses DSLUCS, a Lanczos-type bi-conjugate gradient solver, with incomplete LU-factorization as preconditioner. Users need to beware that iterative methods can be quite “fickle” and may fail for matrices with special features, such as many zeros on the main diagonal, large

numerical range of matrix elements, and nearly linearly dependent rows or columns. Depending on features of the problem at hand, appropriate matrix preconditioning may be essential to achieve convergence. Poor accuracy of the linear equation solution will lead to deteriorated convergence rates for the Newtonian iteration, and an increase in the number of iterations for a given time step. In severe cases, time steps may fail with residuals either stagnating or wildly fluctuating. Information on the convergence of the linear equation solution is written to disk file **LINEQ**, which may be examined with any text editor. Users experiencing difficulties with the default settings are encouraged to experiment with the various solvers and preconditioners included in the TOUGH+ package.

11. Output Specifications

In this section, the various primary and secondary variables that may be provided as output from TOUGH+ are discussed. In addition, data blocks are described for specifying output data, such as the times at which data are printed (block **TIMES**), a list of elements for which output data will be printed (**ELEM_TIME_SERIES** or **FOFT**), a list of connections for which data will be printed (**CONX_TIME_SERIES** or **COFT**), a list of sinks and sources for which data will be printed (**SS_TIME_SERIES** or **GOFT**), and whether to suppress version information (**NOVER**). All these TOUGH+ output capabilities were available in the TOUGH2 family of codes [*Pruess et al*., 1991]. In addition, this section discusses new TOUGH+ output capabilities that provide data describing the evolution of properties and conditions within a designated subdomain (subset) of the simulated domain (grid). This section also includes a few comments on error messages and warnings in the TOUGH+ family of codes.

11.1 Output of Primary and Secondary Variables

The **T+H** code can provide the following output:

1. The pressure, temperatures, saturations, and equilibrium pressure distributions.
2. The mass fractions of the various components in the various phases.
3. Fluxes (in kg/s) and velocities (in m/s) of the phases across the gridblock interfaces (connections) of the domain.
4. The primary variables and their changes in the elements of the domain.
5. Capillary pressures and relative permeabilities.
6. Densities, viscosities (when mobile), and enthalpies of the various phases.
7. Dissociation reaction rates and the corresponding heat of dissociation.
8. Volume and mass balances of the phases and components in the domain.
9. Production rates and production composition at wells.
10. Temporal variations of the most important variables at user-specified elements, connections and sources/sinks.

Of those possible outputs, (1), (8), (9), and (10) are always printed in the standard **T+H** output. The amount of the output is controlled by the parameter `OutputOption` in the data block `PARAM`. In keeping with the TOUGH2 convention, if the `OutputOption` values are increased by 10, printouts will occur after each iteration (not just after convergence).

T+H also provides an output of the evolution of the gas hydrate reaction in the entire simulation domain during the simulation. The data are written to a file named **`Hydrate_Info`**, and include (a) time [days], (b) mass rate of gas release or reaction [kg s^{-1}], (c) volumetric rate of gas release or reaction [standard m^3s^{-1}], (d) cumulative mass of the released gas [kg], (e) cumulative volume of the released gas [standard m^3], (f) cumulative volume of the free gas in the domain [standard m^3], and (g) remaining mass of hydrate in the reservoir [kg].

11.2. Data Block TIMES

This optional block permits the user to obtain printout at specified times. This printout will occur in addition to printout specified in record **PARAM.1**.

Record TIMES.1

Format (2I5, 2E10.4)

```
NumPrintTimes, Max_NumPrintTimes,
TimeStepMax_PostPrint, PrintTimeIncrement
```

NumPrintTimes	The number of times provided on records TIMES.2, TIMES.3, etc., (see below; restriction: NumPrintTimes ≤ 100).
Max_NumPrintTimes	The total number of times at which an output is desired (NumPrintTimes ≤ Max_NumPrintTimes ≤ 100; default is NumPrintTimes = Max_NumPrintTimes).
TimeStepMax_PostPrint	The maximum time step size after any of the prescribed times have been reached (default is infinite).
PrintTimeIncrement	Time increment for times with indices NumPrintTimes, NumPrintTimes+1, ..., Max_NumPrintTimes.

Record TIMES.2, TIMES.3, etc.

Format (8E10.4)

```
(PrintTime(i), i = 1, NumPrintTimes)
```

PrintTime(i)	A list of times (in ascending order) at which printout is desired.

11.3. Data Block `Elem_Time_Series` or `FOFT`

This optional block introduces a list of elements (grid blocks) for which time-dependent data are to be written out for plotting to a file called **`Elem_Time_Series`** during the simulation. If the keyword `'Elem_Time_Series'` (or `'FOFT'`, thus maintaining compatibility with older input files developed for the TOUGH2 family of codes) is present in the input file, then the following data are read:

Record `ELEM_SERIES.1`

Format: (A5, 5X, A3) for `ElemNameLength = 5`, or
(A8, 2X, A3) for `ElemNameLength = 8`

`ObsElemName(1), sum_flag`

`ObsElemName(1)`	The name of the first observation element. If `ObsElemName(1) = '*ALL*'`, then the output describes the evolution of mass and heat balance over the entire domain during the simulation period.
`sum_flag`	If `sum_flag = 'sum'`, then the output describes the evolution of mass and heat balance over the specified subdomain (composed of the elements `ObsElemName(i)` listed in this data block)

Repeat for up to 100 elements (max{`i`} = `100`), entering `ObsElemName(i)` and `sum_flag` in each record.

Record `ELEM_SERIES.2`

Format: (A5) for `ElemNameLength = 5`, or
(A8) for `ElemNameLength = 8`

`(ObsElemName(i), i=2,3,4,…)`

`ObsElemName(i)`	The name of the observation element. Repeat for up to 100 elements (max{`i`} = `100`).

Record `ELEM_SERIES.3`

A blank record closes the **`Elem_Time_Series`** data block.

11.4. Data Block `Conx_Time_Series` or `COFT`

This optional data block introduces a list of connections for which time-dependent data are to be written out for plotting to a file called **`Conx_Time_Series`** during the simulation. If the keyword `'`**`Conx`**`_Time_Series'` (or `'COFT'`, thus maintaining compatibility with older TOUGH2 input files) is present in the input file, then the following data are read:

Record `CONX_SERIES.1`

Format: (A10, 5X, A3) for `ElemNameLength = 5`, or
(A16, 4X, A3) for `ElemNameLength = 8`

`ObsConxName(1), sum_flag`

`ObsConxName(1)`	The name (= the ordered pair of the two constituent element names, i.e., the concatenation `ConxName1// ConxName2` – see Section 7.3) of the first connection to be monitored. If `ObsConxName(1) = '>>`**`ALL`**`<<'`, then the output describes the evolution of mass and heat flows over the entire domain during the simulation period.
`sum_flag`	If `sum_flag = 'sum'`, then the output describes the evolution of the sum of flows through the specified interface (defined by the list of connections in this data block)

Record `CONX_SERIES.2`

Format: (A10, 5X, A3) for `ElemNameLength = 5`, or
(A16, 4X, A3) for `ElemNameLength = 8`

`(ObsConxName(k), k = 2,3,4,…)`

`ObsConxName(i)` The name of the connection to be monitored. Repeat for up to 100 connections (max{`k`} = `100`).

Record `CONX SERIES.3`

A blank record closes the **`Elem_Time_Series`** data block.

The nine columns in the **`Conx_Time_Series_1T`** output file represent the following parameters describing mass and heat flows between the specified connections:

1. Time [sec]
2. Gas mass flow rate [kg/s]
3. Aqueous phase mass flow rate [kg/s]
4. Total heat transfer rate [W]
5. Volumetric flow rate of CH_4 in the gas phase [standard m^3/s]
6. Volumetric flow rate of CH_4 [standard m^3/s]
7. Cumulative volume of produced CH_4 [standard m^3]
8. Cumulative mass of produced H_2O [in kg]
9. Mass fraction of inhibitor in the flow (production) stream

The data in columns 2 to 8 represent cumulative quantities across the list of connections provided in the data block **`Conx_Time_Series`** (or **`COFT`**). If one of the cells in the connection represents a constant-pressure boundary (i.e., in the case of constant-pressure production), then the data in this file provide a means of evaluation of the gas production potential of a hydrate deposit. Note that the data in the **`Conx_Time_Series`** file refer to gas production regardless of origin because it is not possible to determine the provenance of the gas flowing across these interfaces and to reliably differentiate between gas from hydrate dissociation and pre-existing "free" gas in the reservoir.

11.5. Data Block SS_Time_Series or GOFT

This optional data block introduces a list of sinks/sources for which time-dependent data are to be written out for plotting to a file called **SS_Time_Series** during the simulation. If the keyword 'SS_Time_Series' (or 'GOFT', to allow backward compatibility with older TOUGH2 input files) is present in the input file, then the following data are read:

Record SS_SERIES.1

Format (A5)
(ObsSSName(i), i=1,2,3,…)

ObsSSName(i)	Name of the sink or source to be monitored. This should be among one of the names SS_name in the **GENER** data block – see Section 9.1. Repeat for up to 100 sinks/sources (max{i} = 100), one per record. When no sinks or sources are specified here, by default tabulation will be made for all.

Record SS_SERIES.2

A blank record closes the **SS_Time_Series** data block.

The eight columns in the **SS_Time_Series_1T** output file represent the following parameters describing mass and heat flows through the sources or sinks:

1. Time [sec]
2. CH_4 mass flow rate of the sources/sinks [kg/s]
3. CH_4 volumetric flow rate of the sources/sinks [standard m^3/s]
4. Cumulative mass of produced CH_4 [kg]
5. Cumulative volume of produced CH_4 [standard m^3]
6. Water mass flow rate of the sources/sinks [kg/s]
7. Cumulative mass of produced H_2O [kg]
8. Mass fraction of inhibitor in the production stream

When sinks (i.e., wells) are involved, these can provide a means for the evaluation of the gas production potential of a hydrate deposit.

11.6. Data Block SUBDOMAINS

This optional data block is a new TOUGH+ feature. It allows monitoring the evolution of the pore volume-averaged properties and conditions in a set of subdomains (subsets) of the global grid (domain). The time series results are written in a number of separate output files that are equal to the number of the subdomains, and are named according to the convention **xxxxx_Time_Series**, where **'xxxxx'** is the name of the subdomain under observation. If the keyword 'SUBDOMAINS' is present in the input file, then the following data are read using NAMELIST formats:

Record SUBDOMAIN.1

This record includes general data describing the number of subdomains to be monitored and the frequency of recording the results. The namelist in this record is named Real_Parameters_To_Update, and includes the following parameters:

Subdomain_General_Info: An integer variable describing the number of subdomains on which time series data are to be obtained. There is no upper limit in the number of subdomains that can be defined.

NOTE: *Although, technically-speaking, namelists allow easy omission of any parameter within it, the* Subdomain_General_Info *variable must be set to a positive value for the simulation to continue.*

subdomain_output_frequency: An integer variable specifying the recording frequency of the subdomain data in the time series. The default value is 1.

The structure of the `Real_Parameters_To_Update` namelist (and its use as an input format in the data block **`SUBDOMAIN`**) is best illustrated in the example of **Figure 11.1**.

Record `SUBDOMAIN.2`

The namelist in this record provides general information on the subdomain. It is named `Individual_Subdomain_Specifics` and includes the following parameters:

`subdomain_name` The subdomain name is described by this character variable of a maximum of 8 characters. Upon reading the variable, the TOUGH+ code creates an output file named **`xxxxx_Time_Series`**, in `'xxxxx'` is the value of `subdomain_name`. If this variable is omitted in the input, then the TOUGH+ code assigns internally the name `'SubDomNNN'` to `'xxxxx'`, where `'NNN'` is the 3-digit subdomain number in the entry sequence.

`number_of_regions` An integer variable specifying the number of regions of which the subdomain is composed.

NOTE: *The* `number_of_regions` *variable must be set to a positive value for the simulation to continue.*

The structure of the `Individual_Subdomain_Specifics` namelist (and its use as an input format in the data block **`SUBDOMAIN`**) is best illustrated in the example of **Figure 11.1**.

Record `SUBDOMAIN.2.1`

The namelist in this record is named `Region_Specifics`. It provides information that allows the definition of the region within the subdomain and identification of the elements belonging to it, and includes the following parameters:

`definition_mode` This character variable describes the method by which the region is defined. In the current version of TOUGH+, the only available option is `definition_mode = 'Geometry'`, indicating that the region within the subdomain is to be defined through its geometric boundaries.

`region_shape` This character variable describes the shape of the region that is about to be defined. The following self-explanatory options are available:

`= 'Rectangle'` : This option can only be used in Cartesian grids, i.e., if `coordinate_system = 'Rectangle'`.

`= 'Cylinder'` : This option can only be used for either Cartesian or cylindrical grids.

`= 'Sphere'` : This option can only be used for either Cartesian or cylindrical grids.

If `region_shape = 'Rectangle'`, then the following variables must be included in the namelist:

```
Xmin, Xmax, Ymin, Ymax, Zmin, Zmax
```

These real variables indicate the range (minimum and maximum) of the region along the *x*-, *y*- and *z*-axis of the Cartesian coordinate system, respectively.

If `region_shape = 'Cylinder'` and `coordinate_system = 'Cylindrical'`, then the following variables must be included in the namelist:

```
Rmin, Rmax, Zmin, Zmax
```

These real variables indicate the range (minimum and maximum) of the region along the *r*- and *z*-axis of the Cartesiancylindrical coordinate system, respectively.

If `region_shape = ' Cylinder'` and `coordinate_system = 'Cartesian'`, then the following variables must be included in the namelist:

```
Rmin, Rmax,
top_cylinder_center_xyz,
bot_cylinder_center_xyz
```

The real variables `Rmin` and `Rmax` are the minimum and maximum radius of the cylindrical region. The real, 1D arrays `top_cylinder_center_xyz` and `bot_cylinder_center_xyz` have a dimension of 3. These are the (*x,y,z*) coordinates of the centers of the top and bottom circular surfaces of the cylinder, respectively.

If `region_shape = 'Sphere'`, then the following variables must be included in the namelist:

```
Rmin, Rmax, sphere_center_xyz
```

The real variables `Rmin` and `Rmax` are the minimum and maximum radius of the sphere. The (*x,y,z*) coordinates of the center of the sphere are stored in the elements of the real, 1D array `top_cylinder_center_xyz` of dimension 3. Of those three values, only the 3rd (corresponding to the *z*-coordinate of the center of the sphere) one is used when `coordinate_system = 'Cylindrical'`.

Thus, its is possible to combine regions of very different geometries to create subdomains that are very irregular in shape.

NOTE #1: *All the various variable combinations and permutations in namelist* `Region_Specifics` *must be defined for the simulation to continue.*

NOTE #2: *All the regions within the subdomain must be defined. A total of* `number_of_regions` *records* `SUBDOMAIN.2.1` *(one for each region of the subdomain) must be provided. Thus, if we define two subdomains, the first with 3 regions and the second with 4, then we need to provide the following data: For the 1st Subdomain: One* `SUBDOMAIN.2` *record, and 3* `SUBDOMAIN.2.1` *records. For the 2nd Subdomain: One* `SUBDOMAIN.2` *record, and 4* `SUBDOMAIN.2.1` *records.*

The structure and use of the `Region_Specifics` namelist in the data block **`SUBDOMAIN`** is best illustrated by the example of **Figure 11.1**.

```
.
.
.
SUBDOMAINS
&Subdomain_General_Info  number_of_subdomains       = 1,
                         subdomain_output_frequency = 1
                         /
   &Individual_Subdomain_Specifics  subdomain_name = 'Zone1',
                                    number_of_regions = 1
                                    /
      &Region_Specifics  definition_mode = 'Geometry',
                         region_shape    = 'Rectangle',   ! Can be <rectangle>, <sphere> or <cylinder>
                         Xmin =  2.0e-1, Xmax = 4.0e-1,   ! Range (min and max) along the first
coordinate axis
                         Ymin = -1.0e8,  Ymax = 1.0e8,    ! Range (min and max) along the second
coordinate axis
                         Zmin = -1.0e8,  Zmax = 1.0e8,    ! Range (min and max) along the third
coordinate axis
                       ! Rmin, Rmax,                      ! Range (min and max) of the spherical or
cylindrical subdomain
                       ! top_cylinder_center_xyz,         ! Coordinates of the center of the top
surface of a cylindrical region
                       ! bot_cylinder_center_xyz,         ! Coordinates of the center of the bottom
surface of a cylindrical region
                         /
```

Figure 11.1. An example of the **`SUBDOMAINS`** data block for tracking the evolution of volume-averaged properties and conditions in specified subdomains. This data block uses namelist-based formats for data inputs.

11.7. Data Block NOVER

One record with **NOVER** typed in columns 1-5 will suppress printing of a summary of versions and dates of the program units used in a TOUGH+ run.

11.8. Warning Output and Error Messages

If inputs indicate conflicting conditions and/or parameter values are outside realistic ranges, TOUGH+ is designed to respond according to the severity of the violation. Non-critical conflicts result in a warning or clarifying message, internal adjustment of the corresponding conditions and/or parameters, and continuation of the simulation. Serious violations result in an error message identifying the problem, and the simulation is aborted.

12.0. Example Problems

12.1. Example Files and Naming Conventions

The files corresponding to the examples discussed in this manual can be found in the directory `"TestRuns"` in the CD accompanying this manual. The input files of the example problems have the general name `Test_XXX` (where `XXX` is an identifier) and are the following:

1. `Test_1T`
2. `Test_1Tk`
3. `Test_1P`
4. `Test_1Pk`
5. `Test_1P_ice`
6. `Test_1TbS`
7. `Test_1TbSk`
8. `Test_1TSk`
9. `Test_2Qp`
10. `Test_3T`
11. `Test_3Qp`
12. `Test_3Qpk`
13. `Test_RadH1`
14. `Test_2D`

The corresponding output files are also included in the directory `"TestRuns"`. The naming convention of the generic TOUGH+ output files involves the suffix `".out"` at the end of the input file name. The hydrate-specific output files have the general name `Hydrate_Info_XXX`, where `XXX` is the identifier of the input file name.

For some examples in which the hydration reaction (dissociation or formation) is induced by boundary conditions, the additional output files of the time series at the boundary connection are included under the general name **`Conx_Time_Series_XXX`**. Similarly, if the hydration reaction is induced by sources or sinks, the corresponding additional output file of the time series of the conditions at the sinks and/or sources is included in the directory `"TestRuns"` under the name **`SS_Time_Series_XXX`**.

12.2. Problem Test_1T: Thermal Stimulation, Equilibrium Dissociation, No Inhibitor

This 1-D problem is designed to demonstrate the basic concepts of hydrate dissociation by means of thermal stimulation. Note that it is practically impossible to separate the thermal and pressure processes in the course of dissociation, regardless of the dissociation method. As hydrate dissociates in thermal stimulation, gas evolves and accumulates in porous media, leading to pressure increases immediately ahead of the hydrate interface that shift the hydrate toward the stability zone. Conversely, depressurization is a very effective method of hydrate dissociation, the endothermic nature of which results in rapid temperature drop and enhanced stability of the remaining hydrate.

For convenience, the input file is listed in **Figure 12.1**. As an exercise, a novice user is urged to identify the various variables and parameters in the input file.

The 1-m long domain (represented by 10 active cells of uniform Δx = 0.1 m size) is a pressurized and thermally insulated column of a porous medium, in which hydrate and water coexist at a pressure of 4.0×10^6 Pa and T = 1.2 ^{o}C, i.e., well within the hydrate stability zone. At a time t = 0, the boundary at x = 0 becomes permeable and the column comes in contact with warm water at a temperature of T_b = 45 ^{o}C. As the temperature in the column increases, hydrate dissociation is expected to occur, leading to the evolution of gas, which will then escape the column by flowing through the high-temperature boundary. Note that the pressure in the high-temperature boundary is equal to the initial one in the column, thus allowing escape of the pressurized gas. In this simulation, the hydrate dissociation reaction is assumed to occur at equilibrium. Under these conditions, dissociation is limited and controlled by heat transfer issues.

The porous medium has a porosity ϕ = 0.3, and a permeability $k = 2.96 \times 10^{-13}$ m^2 (= 30 mD in oilfield units). In the presence of the ice and hydrate solid phases, the critical mobile porosity (i.e., the "free" porosity -- not occupied by solids -- below which the porous medium becomes impermeable) is 0.05, and the porosity reduction exponent is 3.

Note that a non-zero (and relatively significant) pore compressibility (=10^{-8} 1/Pa) is assigned to the porous medium. This is necessary in hydrate simulations, in which evolution of solid phases of lower density (such as ice and hydrate) can lead to extraordinarily high pressures as the aqueous phase disappears if pore compressibility is small. In this illustrative example, the thermal conductivity (=3.1 W/m/K) is relatively

large, but was given this value to enhance heat conduction (the main heat transfer mechanism from the high-temperature boundary).

The hydrate properties and the type of the reaction are listed in the `HYDRATE` data block. The thermal conductivity, specific heat and density of the CH_4 hydrate are from data in *Sloan* [1998], and are constants because no information is available on their dependence on temperature and/or pressure. However, **T+H** assumes that the constant input density of the CH_4 hydrate is that at the quadruple point, and the hydrate density in the simulations is internally adjusted by assuming that its compressibility and thermal expansivity are the same as those of ice (as calculated within TOUGH+). The inhibitor data correspond to the properties of NaCl.

`Test_1T.out`, the standard TOUGH+ output corresponding to the input file **`Test_1T`**, can be found in the directory `"TestRuns"` in the CD accompanying this manual. Because `MOP(5) = 3`, the output includes detailed messages about the evolution of the residuals during the Newtonian iterations at each time step, as well as about phase changes occurring during the course of the simulations. Because `OutputOption = 3`, a full output is obtained that provides a very detailed list of the conditions, parameters and thermophysical properties of the system at each cell and at each connection. Thus, the output describes the pressure, temperature, phase saturation, partial CH_4, H_2O-vapor pressure, equilibrium hydration pressure, inhibitor mass fraction in the aqueous phase, CH_4 concentrations in the aqueous and gas phases, phase densities, porosity, capillary pressure, relative permeability of the gas and aqueous phases, heat and fluid fluxes, mobile phase velocities, CH_4 fluxes in the aqueous and gas phases, primary variables and their changes, phase enthalpies, rate of the hydration reaction, heat of the

hydration reaction, and the temperature shift in the hydrate P-T equilibrium caused by the presence of the inhibitor. Additionally, the output provides mass and volume balances of the phases, component mass balances, and component distribution into the phases.

The results in the portion of the output in the **`Test_1T.out`** file (see Appendix A) are consistent with the expected system response. The hydrate dissociation front is observed to move deeper into the column with time, and is in the eighth cell at the end of the simulation period (55th timestep). As temperature increases, hydrate begins to dissociate (see the gradual hydrate saturation decline and eventual disappearance in the warming cells, along with an emerging and then increasing gas saturation). The maximum pressure is observed at the dissociation front, indicated by the first incidence of coexistence of aqueous, gas and hydrate phases in a cell (and immediately ahead of a cell from which all hydrate has disappeared). This higher pressure is caused by the gas evolution in response to hydrate dissociation, and leads to formation of additional hydrate (indicated by higher hydrate saturations and a positive hydrate reaction rate) in the cell immediately in front of the one that contains the dissociation front. As expected, the gas fluxes indicate gas flow toward the high-temperature permeable boundary. Note that, because the maximum pressure occurs at the hydrate dissociation front, gas flows in both directions from the cells that contain the hydrate interface. Comparison of the partial pressure of CH_4 to the hydrate equilibrium pressure confirms that dissociation occurs at equilibrium.

The additional output file **`Hydrate_Info_1T`** (also listed in Appendix A) provides a measure of the cumulative dissociation behavior in the entire system, as opposed to that in individual cells (described in the standard TOUGH+ output). This output file features

self-explanatory headings, and includes the instantaneous cumulative rates of CH_4 gas evolution from hydrate (expressed as mass rates or standard volume rates), as well as the corresponding cumulative amount (in terms of total mass and total standard volume) of hydrate-originating CH4. Thus, it provides a measure of the overall performance of the hydrate-bearing system as a gas production source. Negative rates and masses/volumes indicate hydrate generation. Note that the results in any **`Hydrate_Info`** output indicate conditions and response within the accumulations, and do not necessarily reflect behavior and performance at the production well. Thus, for successful exploitation of a hydrate accumulation, significant CH4 volumes must be released through dissociation within the deposit, but the gas must also be able to be removed from the system through an appropriate collection (i.e., well) facility.

```
"Test1" ... Hydrate dissociation, thermal stimulation
MEMORY
'HYDRATE-EQUILIBRIUM'
   2     3    4    .FALSE.          ! NumCom, NumEqu, NumPhases, binary_diffusion
'Cartesian' 15 30 5 .FALSE. .FALSE. ! coordinate_system,Max_NumElem,Max_NumConx,ElemNameLength,active_conx_only,
   2                                ! MaxNum_SS
   2                                ! MaxNum_Media
.FALSE.  .FALSE.  .FALSE.        ! element_by_element_properties, porosity_perm_dependence, scaled_capillary_pressure
.FALSE.  'Continuous'            ! coupled_geochemistry, property_update [= 'Continuous', 'Iteration', 'Timestep']
.FALSE.  ' '  'Continuous'  0    ! coupled_geomechanics, geomechanical_code_name, property_update, num_geomech_param
ROCKS----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
DIRT1    1     2.6e3       .30  2.96E-13  2.96E-13  2.96E-13       3.1     1000.
     1.e-8              0.5e0                                  5.0e-2     3.0e0
BOUND    0     2.6e3     0.0e0  0.00E-13  0.00E-13  0.00E-13     1.0e2     1000.

HYDRATE--1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
     1                       ! HCom%NCom
'CH4'  6.0d0 1.00d00         ! Name, hydration number, mole fraction in composite hydrate
     1                       ! Number of coefficients in thermal conductivity polynomial
  4.5e-1                     ! Coefficients in the thermal conductivity polynomial
     1                       ! Number of coefficients in the specific heat polynomial
  2.1e03                     ! Coefficients in the specific heat polynomial
     1                       ! Number of coefficients in density polynomial
  9.2e02                     ! Coefficients in the density polynomial
.FALSE.                      ! inhibitor_present,T_MaxOff,C_MaxOff,MW_Inhib,D_Inhib,H_InhSol,DifCo_Inh
2                            ! F_EqOption
'EQUILIBRIUM'                ! Type of dissociation
START----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
----*----1 MOP: 123456789*123456789*1234 ---*----5----*----6----*----7----*----8
PARAM----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
   3 080     0101000300400200040000311 1   0.00E-5
            2.200E+5     1.0e2   8.64E+4              9.8060
     1.E-5     1.E00                                  1.0e-8            AqH
             4.000e6               0.5e0                1.20
ELEME
A00 1              10.1000E+000.4000E+00          0.5100E-01-.5000E+00-.5000E+00
A00 2              10.1000E+000.4000E+00          0.1510E+00-.5000E+00-.5000E+00
A00 3              10.1000E+000.4000E+00          0.2510E+00-.5000E+00-.5000E+00
A00 4              10.1000E+000.4000E+00          0.3510E+00-.5000E+00-.5000E+00
A00 5              10.1000E+000.4000E+00          0.4510E+00-.5000E+00-.5000E+00
A00 6              10.1000E+000.4000E+00          0.5510E+00-.5000E+00-.5000E+00
A00 7              10.1000E+000.4000E+00          0.6510E+00-.5000E+00-.5000E+00
A00 8              10.1000E+000.4000E+00          0.7510E+00-.5000E+00-.5000E+00
A00 9              10.1000E+000.4000E+00          0.8510E+00-.5000E+00-.5000E+00
A0010              10.1000E+000.4000E+00          0.9510E+00-.5000E+00-.5000E+00
ina
A00 0              10.1000E-020.4000E-02          0.5000E-03-.5000E+00-.5000E+00

CONNE
A00 0A00 1                   10.5000E-030.5000E-010.1000E+01
A00 1A00 2                   10.5000E-010.5000E-010.1000E+01
A00 2A00 3                   10.5000E-010.5000E-010.1000E+01
A00 3A00 4                   10.5000E-010.5000E-010.1000E+01
A00 4A00 5                   10.5000E-010.5000E-010.1000E+01
A00 5A00 6                   10.5000E-010.5000E-010.1000E+01
A00 6A00 7                   10.5000E-010.5000E-010.1000E+01
A00 7A00 8                   10.5000E-010.5000E-010.1000E+01
A00 8A00 9                   10.5000E-010.5000E-010.1000E+01
A00 9A0010                   10.5000E-010.5000E-010.1000E+01
A0010A0011                   10.5000E-010.5000E-030.1000E+01

RPCAP----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
    9           .120       .02        3.
    7        0.45000   1.10e-1   8.0E-05      1.E6     1.0e0
COFT -----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
A00 0A00 1


GENER


INCON----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
A00 0           0.30000000E+00  Aqu
             4.000e6              00.0e0               45.00


ENDCY----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
```

Figure 12.1. Input file for example problem Test_1T (in Section 12.2) involving thermal stimulation, equilibrium dissociation, and no inhibitor.

12.3. Problem Test_1Tk: Thermal Stimulation, Kinetic Dissociation, No Inhibitor

This 1-D problem is entirely analogous to that discussed in `Test_1T`, from which it differs only in that hydrate dissociation is not treated as an equilibrium reaction but as a kinetic reaction. For convenience, the input file **`Test_1Tk`** is listed in **Figure 12.2**. It can be seen that the inputs are identical, with the exception of parameters and values describing the kinetic nature of the reaction.

Because a kinetic reaction confers an additional degree of freedom, the number of mass components `NumCom` increases by one to `NumCom = 3` (the hydrate is now a mass component in addition to being a solid phase, see discussion in Section 2.2). Consequently, the number of equations per cell increases by one to `NumEqu = 4`, the number of primary variables describing initial conditions (in data blocks `PARAM` and **`INCON`**) increases by one, and the corresponding Jacobian matrix is larger than the one in problem **`Test_1T`**. As an exercise, a novice user is urged to identify the primary variables in this input file, and to differentiate them from those in the input file **`Test_1T`**.

Both the `EOS_Name` heading in the `MEMORY` block (record `MEMORY.2)`, and in the `Reaction_Type` variable in the `HYDRATE` data block (record `HYDRATE.11)` indicate that this is kinetic system. The values of the kinetic parameters of the hydration reaction (i.e., intrinsic reaction rate and the activation energy) were obtained from *Kim et al.* (1987), and *Clarke and Bishnoi* (2002). The area adjustment factor (variable `Area_Factor` in record `HYDRATE.12` (also see equation 2.6) was assumed to be equal to one.

The interested user is directed to the directory "TestRuns" for review of the standard TOUGH+ output, as well as of the additional **Hydrate_Info_1Tk** and **Conx_Time_Series_1Tk** output files. Upon inspection, a general observation that can be made is that this is a more difficult problem to solve, and that it takes more timesteps (65 vs. 54) to cover the simulation period than in problem **Test_1T**. A reason for the slower time advance is the kinetic hydrate dissociation, which imposes limits on the time step size.

The results of the illustrative problems in **Test_1T** and **Test_1Tk** are compared in **Figures 12.3** to **12.6**. It should be noted that the results of these two simulations are affected by the relatively coarse discretization in the two simple systems, but conclusions drawn from the relative performance are valid and provide a robust perspective in the evaluation of such simulation results.

Figure 12.3 shows the cumulative rates of CH_4 release from the dissociation of hydrate in the columns of problems **Test_1T** and **Test_1Tk**. The effect of the relatively coarse discretization is evident in the oscillatory nature of the curves of rate evolution over time. An increase in the rate of CH_4 release is a direct consequence of faster dissociation, which leads to a localized increase in pressure (caused by the evolving gas and the resistance to flow posed by the porous medium) and a drop in temperature (caused by the endothermic nature of the hydrate reaction). The increase in pressure and temperature drop result in a subsequent reduction in the rate of dissociation until conduction and convection (of the advancing warm water) can raise the temperature and accelerate dissociation. The result of this sequence of events is the (roughly) periodic nature of the rate curve (involving the repetition of a similar pattern), which is

substantially affected by the size of the cells. Thus, each of the four distinct patterns in **Figure 12.3** corresponds to a different cell, and represents the advance of the hydrate dissociation front through that cell at the time of observation.

A significant observation that can be drawn from **Figure 12.3** is that the equilibrium and kinetic rates are different (with the kinetic rate lower) when the dissociation front first moves into a cell, but practically coincide afterwards. However, at the very beginning of the simulation, a burst of very rapid gas release is observed in the case of kinetic dissociation. This pattern of behavior is consistent with expectations because the maximum deviation between kinetic and equilibrium predictions is anticipated to occur at the maximum deviation from equilibrium. This is followed by equilibrium dissociation along the L+G+W phase boundary. The coincidence of the kinetic and equilibrium rates indicates that equilibrium is established very rapidly, as dictated by the relatively fast intrinsic dissociation rate and the high temperature. In cases of thermal stimulation involving high temperatures, equilibrium and kinetic dissociation models are expected to give very similar results (in terms of overall gas release from the hydrate) because (a) the high intrinsic reaction rate does not pose a kinetic limitation on the reaction, and (b) the reaction rate is dominated by heat transfer issues.

In **Figure 12.4**, the cumulative gas volumes released from the hydrate in the two problems are very similar, and confirm the observations from the discussion on the rates. The discretization effects are evident in the distinctly "segmented" appearance of the curves, which exhibit "bumps" corresponding to the periodic rate patterns in **Figure 12.3**. Note that **Figures 12.3** and **12.4** were plotted using the data from the output files **`Hydrate_Info_1T`** and **`Hydrate_Info_1Tk`**.

In problems **`Test_1T`** and **`Test_1Tk`**, gas is produced as it escapes into the warm water across the $x = 0$ boundary. **Figures 12.5** and **12.6** (based on data from the **`Conx_Time_Series_1T`** and **`Conx_Time_Series_1Tk`** output files) show the volumetric production rate of CH_4 and the cumulative CH_4 production. In these two examples, all the CH_4 originates from the dissociation of the hydrate because no 'free' CH_4 was initially present in the system. **Figure 12.5** shows the same periodic pattern observed in **Figure 12.3**, denoting the effect of discretization. The CH_4 production rates (i.e., the fluxes across the $x = 0$ boundary) are initially the same, but then the kinetic rate decreases (relative to the equilibrium rate). This difference occurs as the hydrate dissociation front is about to move into an adjacent cell. The cumulative effect is reflected in the total gas production (shown in **Figure 12.4**, and measured as the total CH_4 volume that crosses the $x = 0$ boundary), which indicates lower gas production for kinetic dissociation. The effect of discretization is apparent in the "segmented appearance" of the gas production curves, which correspond directly to the periods in the production rates in **Figure 12.5**.

```
"Test_1Tk" ... Kinetic hydrate dissociation, thermal stimulation
MEMORY
'HYDRATE-KINETIC'
   3     4    4    .FALSE.          ! NumCom, NumEqu, NumPhases, binary_diffusion
'Cartesian' 15 30 5 .FALSE. .FALSE. ! coordinate_system,Max_NumElem,Max_NumConx,ElemNameLength,active_conx_only,
   2                                ! MaxNum_SS
   2                                ! MaxNum_Media
.FALSE.  .FALSE.  .FALSE.     ! element_by_element_properties, porosity_perm_dependence, scaled_capillary_pressure
.FALSE.  'Continuous'         ! coupled_geochemistry, property_update [= 'Continuous', 'Iteration', 'Timestep']
.FALSE.  ' '  'Continuous'  0 ! coupled_geomechanics, geomechanical_code_name, property_update, num_geomech_param
ROCKS----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
DIRT1    1     2.6e3       .30  2.96E-13  2.96E-13  2.96E-13       3.1     1000.
     1.e-8             0.50e0                                  5.0e-2     3.0e0
BOUND    0     2.6e3     0.0e0  0.00E-13  0.00E-13  0.00E-13     1.0e2     1000.

HYDRATE--1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
DIRT1    1     2.6e3       .30  2.96E-13  2.96E-13  2.96E-13       3.1     1000
     1                     ! HCom%NCom
'CH4'  6.0d0 1.00d00       ! Name, hydration number, mole fraction in composite hydrate
     1                     ! Number of coefficients in thermal conductivity polynomial
  4.5e-1                   ! Coefficients in the thermal conductivity polynomial
     1                     ! Number of coefficients in the specific heat polynomial
  2.1e03                   ! Coefficients in the specific heat polynomial
     1                     ! Number of coefficients in density polynomial
  9.2e02                   ! Coefficients in the density polynomial
.FALSE.                    ! inhibitor_present,T_MaxOff,C_MaxOff,MW_Inhib,D_Inhib,H_InhSol,DifCo_Inh
2                          ! F_EqOption
'KINETIC'                  ! Type of dissociation
8.1d4   3.6d4  1.0e-0      ! Activation energy, intrinsic rate constant, area factor
START----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
----*----1 MOP: 123456789*123456789*1234 ---*----5----*----6----*----7----*----8
PARAM----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
   3 080     01010003014002000040003111   0.00E-5
            2.200E+5    1.0e01   8.64E+4              9.8060
     1.E-5     1.E00                                  1.0e-8            AqH
            4.000e6               0.5e0               0.0e0                1.20
ELEME
A00 1              10.1000E+000.4000E+00          0.5100E-01-.5000E+00-.5000E+00
A00 2              10.1000E+000.4000E+00          0.1510E+00-.5000E+00-.5000E+00
A00 3              10.1000E+000.4000E+00          0.2510E+00-.5000E+00-.5000E+00
A00 4              10.1000E+000.4000E+00          0.3510E+00-.5000E+00-.5000E+00
A00 5              10.1000E+000.4000E+00          0.4510E+00-.5000E+00-.5000E+00
A00 6              10.1000E+000.4000E+00          0.5510E+00-.5000E+00-.5000E+00
A00 7              10.1000E+000.4000E+00          0.6510E+00-.5000E+00-.5000E+00
A00 8              10.1000E+000.4000E+00          0.7510E+00-.5000E+00-.5000E+00
A00 9              10.1000E+000.4000E+00          0.8510E+00-.5000E+00-.5000E+00
A0010              10.1000E+000.4000E+00          0.9510E+00-.5000E+00-.5000E+00
ina
A00 0              10.1000E-020.4000E-02          0.5000E-03-.5000E+00-.5000E+00

CONNE
A00 0A00 1                   10.5000E-030.5000E-010.1000E+01
A00 1A00 2                   10.5000E-010.5000E-010.1000E+01
A00 2A00 3                   10.5000E-010.5000E-010.1000E+01
A00 3A00 4                   10.5000E-010.5000E-010.1000E+01
A00 4A00 5                   10.5000E-010.5000E-010.1000E+01
A00 5A00 6                   10.5000E-010.5000E-010.1000E+01
A00 6A00 7                   10.5000E-010.5000E-010.1000E+01
A00 7A00 8                   10.5000E-010.5000E-010.1000E+01
A00 8A00 9                   10.5000E-010.5000E-010.1000E+01
A00 9A0010                   10.5000E-010.5000E-010.1000E+01
A0010A0011                   10.5000E-010.5000E-030.1000E+01

RPCAP----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
    9           .120       .02        3.
    7        0.45000   1.10e-1   8.0E-05      1.E6     1.0e0
COFT -----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
A00 0A00 1


GENER


INCON----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
A00 0           0.30000000E+00  Aqu
            4.000e6               0.0e0               0.0e0               45.00

ENDCY----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
```

Figure 12.2. Input file for example problem Test_1Tk (in Section 12.3) involving thermal stimulation, kinetic dissociation, and no inhibitor.

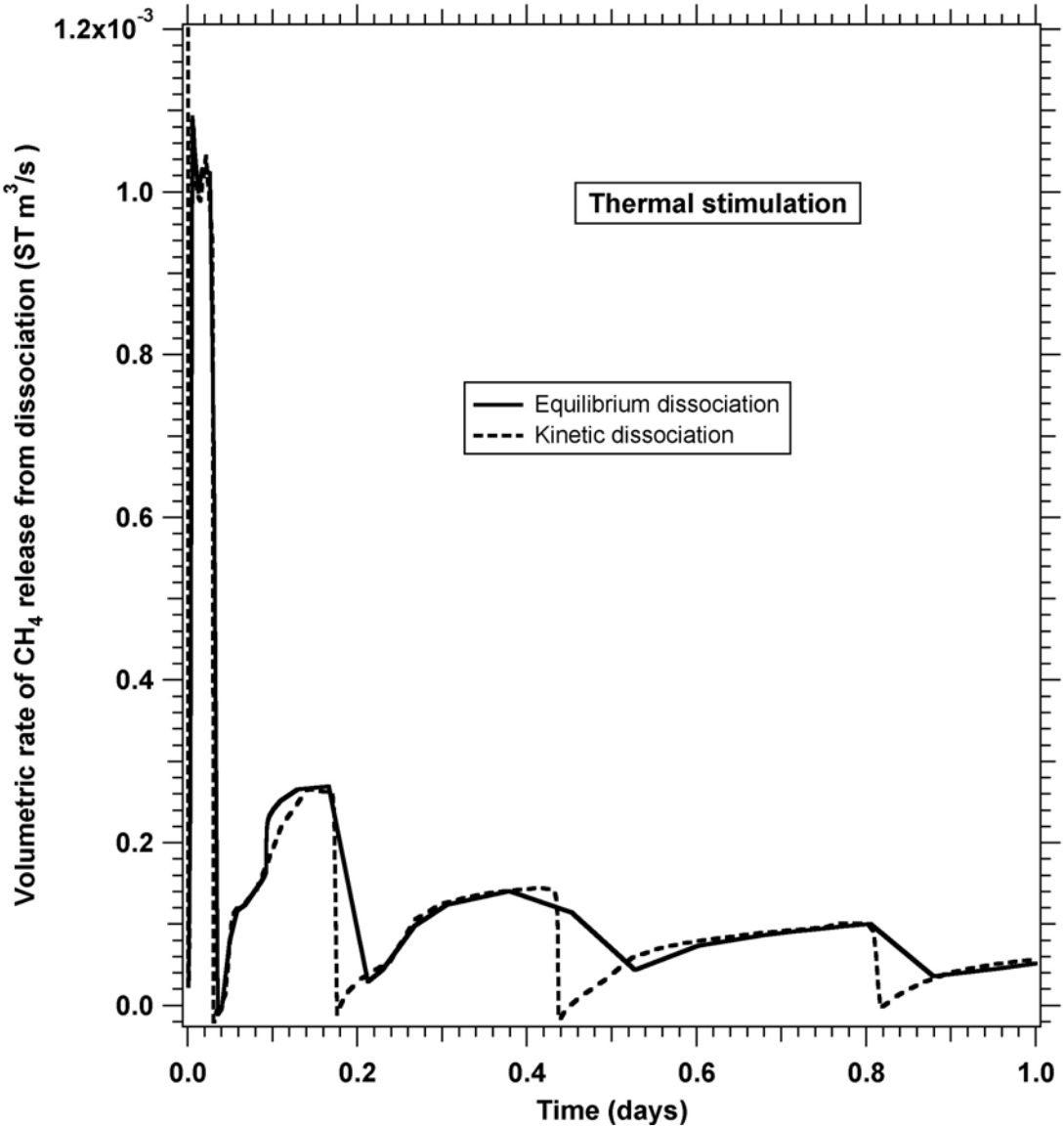

Figure 12.3. Comparison of the volumetric rates of CH_4 release from hydrate dissociation in problems `Test_1T` and `Test_1Tk`.

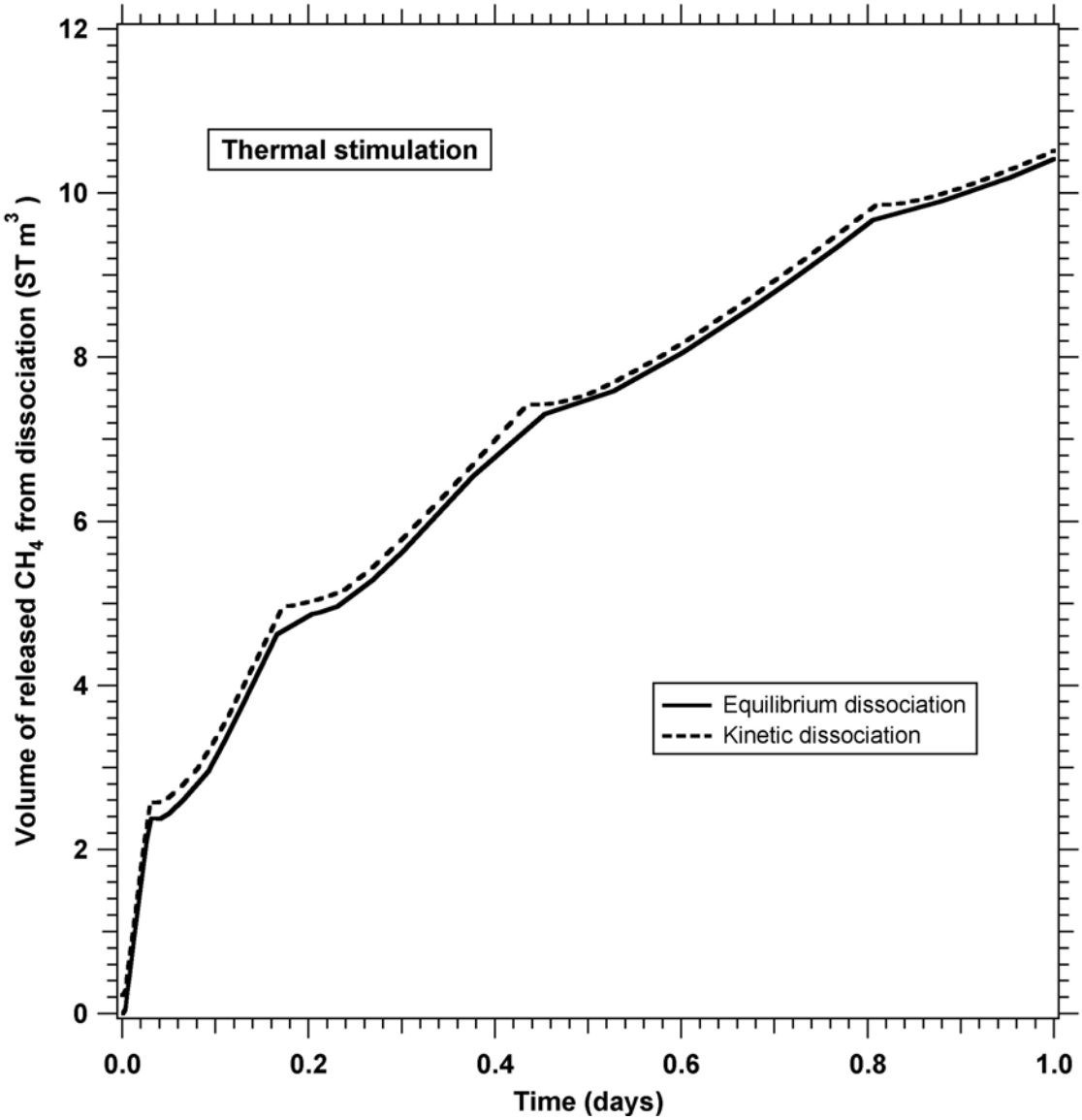

Figure 12.4. Comparison of the cumulative volumes of CH_4 released from hydrate dissociation in problems `Test_1T` and `Test_1Tk`.

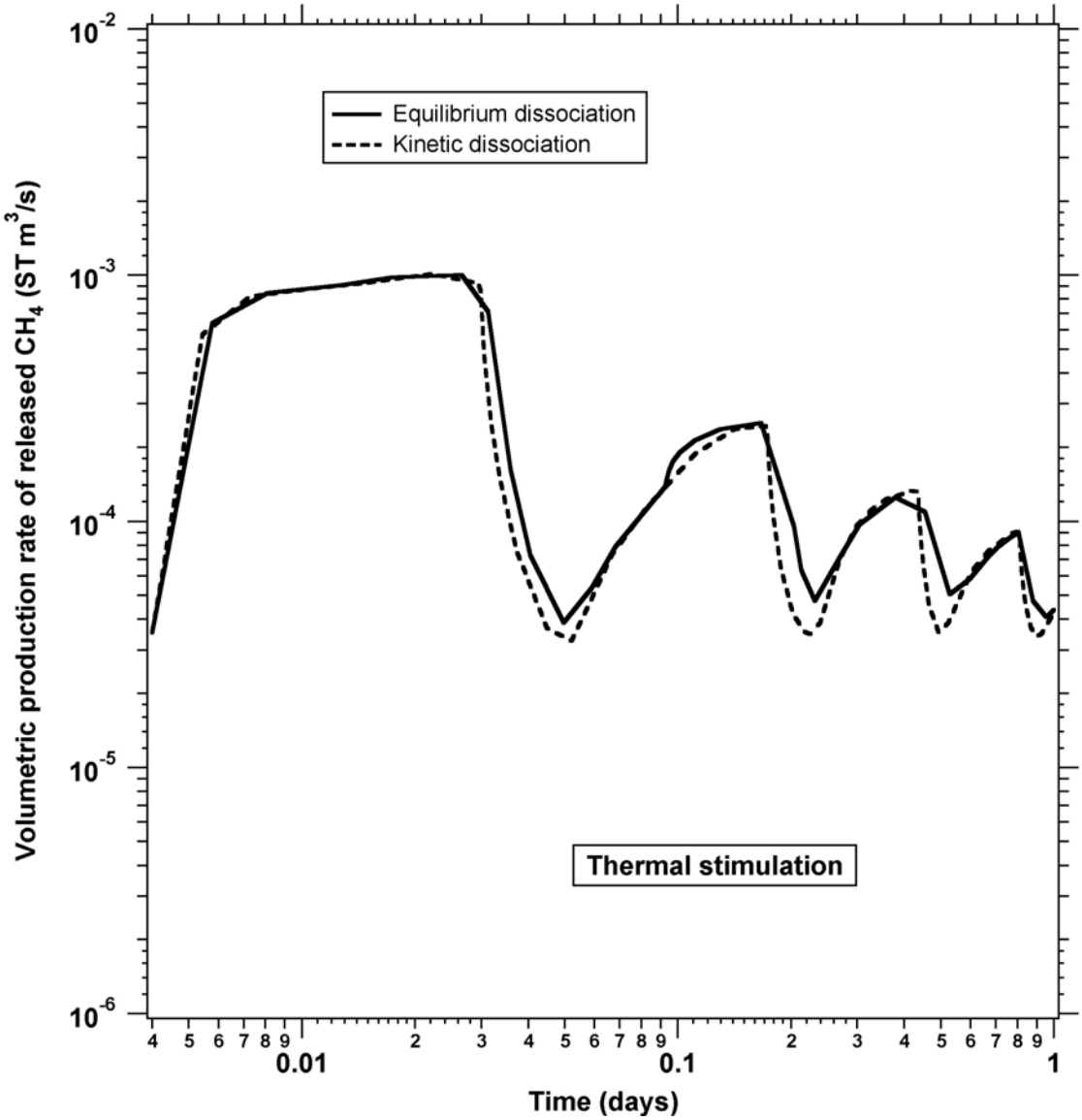

Figure 12.5. Comparison of the volumetric production rates of CH_4 (measured as the flux crossing the x = 0 boundary) in problems `Test_1T` and `Test_1Tk`.

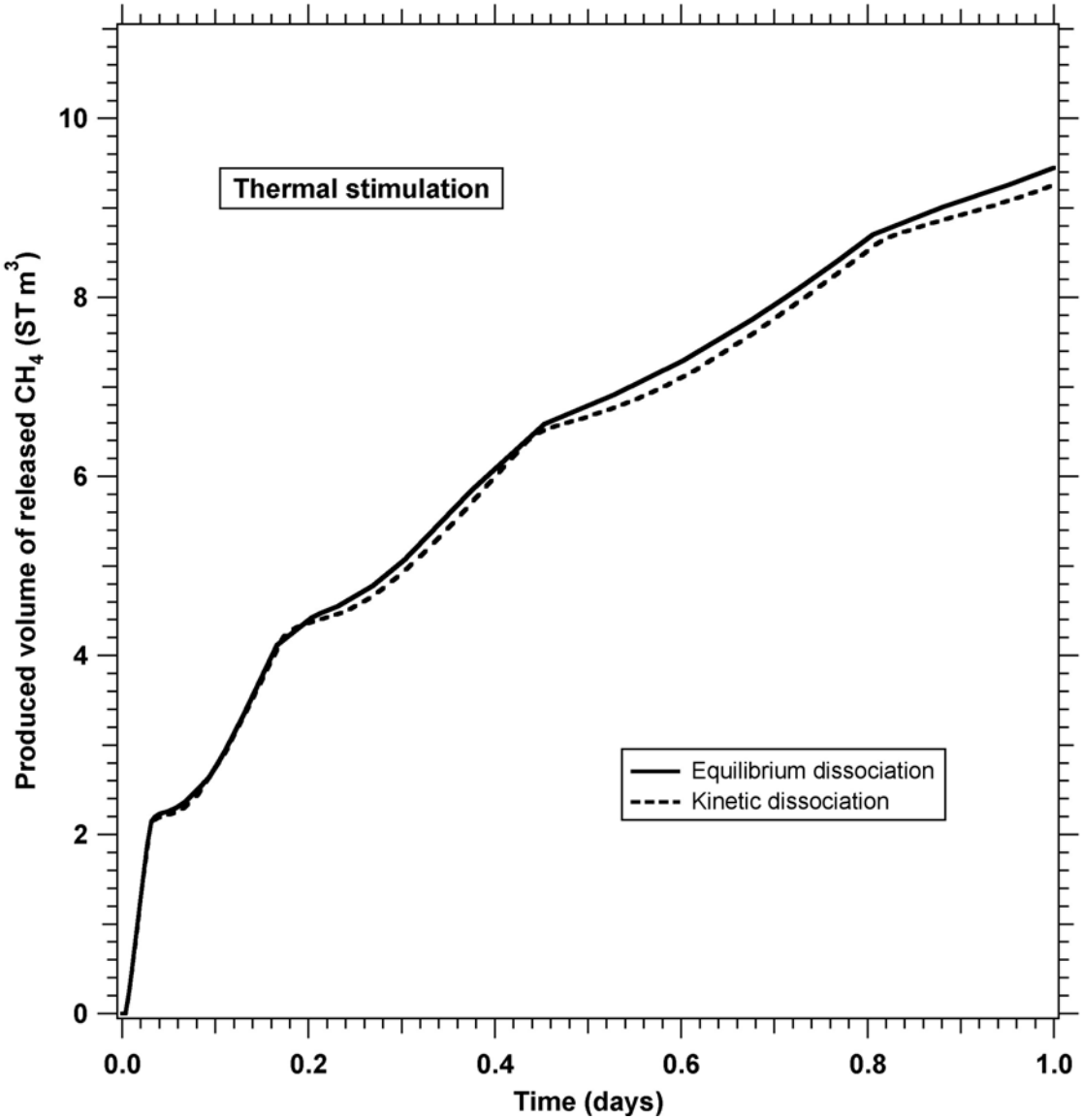

Figure 12.6. Comparison of the total production volumes of CH_4 (measured as the total volumes crossing the x = 0 boundary) in problems `Test_1T` and `Test_1Tk`.

12.4. Problems Test_1P: Depressurization, Equilibrium Dissociation, No Inhibitor

The only difference between the two-phase (hydrate and aqueous) 1-D problem in **`Test_1P`** and that discussed in **`Test_1T`** is in the boundary conditions at x = 0. Instead of a high-temperature permeable boundary (leading to thermal dissociation), the x = 0 boundary is now kept at a pressure of P_b = 2.7x10^6 Pa, which is lower than the initial pressure in the remainder of the column (P = 6x10^6 Pa). The pressure differential between the column and boundary and the non-zero permeability of the connection joining the two allow depressurization, which in turn induces dissociation. Note that the boundary pressure P_b is larger than the pressure at the quadruple point of CH_4-hydrate. This allows dissociation without reaching the quadruple point, which consequently keeps ice from forming. The constant boundary temperature of T_b = 1 ^{o}C is not expected to play a significant role in dissociation, despite its being above the melting point, because heat convection (fueled by fluid flow) is the main means of heat transfer, with conduction being a slow and inefficient process.

In the absence of an external heat source, the strongly endothermic reaction of hydrate dissociation is fueled by the heat provided by its surroundings. Consequently, temperature is expected to decline rapidly as dissociation proceeds.

The input file for the problem **`Test_1P`** appears in **Figure 12.7**. As an exercise, the novice user is asked to identify the differences between this file and the file **`Test_1T`**. The output files corresponding to this simulation (**`Test_1P.out`**, **`Hydrate_Info_1P`** and **`Conx_Time_Series_1P`**) can be found in the directory `"TestRuns"` in the accompanying CD.

Review of the output files indicates that the evolution of temperature during dissociation conforms to expectations, exhibiting sharp and rapid declines. A very important observation is the much faster response of the hydrate to depressurization than to thermal stimulation. This difference in response was expected because of the very significant velocities at which the pressure and the temperature fronts propagate. The inevitable conclusion from these results (discussed more fully in Section 12.6) is that depressurization is a much more efficient hydrate dissociation process than thermal stimulation.

```
"Test_1P" ... Hydrate dissociation - depressurization
MEMORY
'HYDRATE-EQUILIBRIUM'
   2     3    4    .FALSE.           ! NumCom, NumEqu, NumPhases, binary_diffusion
'Cartesian' 15 30 5 .FALSE. .FALSE.  ! coordinate_system, Max_NumElem, Max_NumConx, ElemNameLength, active_conx_only,
   2                                 ! MaxNum_SS
   2                                 ! MaxNum_Media
.FALSE. .FALSE. .FALSE.            ! element_by_element_properties, porosity_perm_dependence, scaled_capillary_pressure
.FALSE.  'Continuous'              ! coupled_geochemistry, property_update [= 'Continuous', 'Iteration', 'Timestep']
.FALSE.  ' '  'Continuous'  0      ! coupled_geomechanics, geomechanical_code_name, property_update, num_geomech_param
ROCKS----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
DIRT1    1     2.6e3       .30  2.96E-13  2.96E-13  2.96E-13       3.1     1000.
     1.e-8             0.50e0                                  5.0e-2     3.0e0
BOUND    0     2.6e3     0.0e0  0.00E-13  0.00E-13  0.00E-13     1.0e2     1000.

HYDRATE--1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
     1                       ! HCom%NCom
'CH4'  6.0d0 1.00d00         ! Name, hydration number, mole fraction in composite hydrate
     1                       ! Number of coefficients in thermal conductivity polynomial
  4.5e-1                     ! Coefficients in the thermal conductivity polynomial
     1                       ! Number of coefficients in the specific heat polynomial
  2.1e03                     ! Coefficients in the specific heat polynomial
     1                       ! Number of coefficients in density polynomial
  9.2e02                     ! Coefficients in the density polynomial
.FALSE.                      ! inhibitor_present,T_MaxOff,C_MaxOff,MW_Inhib,D_Inhib,H_InhSol,DifCo_Inh
2                            ! F_EqOption
'EQUILIBRIUM'                ! Type of dissociation
START----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
----*----1 MOP: 123456789*123456789*1234 ---*----5----*----6----*----7----*----8
PARAM----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
   3 080     00210003004002000400003000   0.00E-5
            8.640E+5    1.0e00   8.64E+6              9.8060
     1.E-5     1.E00                                  1.0e-8            AqH
             6.000e6              5.0e-1                7.20
ELEME
A00 1              10.1000E+000.4000E+00          0.5100E-01-.5000E+00-.5000E+00
A00 2              10.1000E+000.4000E+00          0.1510E+00-.5000E+00-.5000E+00
A00 3              10.1000E+000.4000E+00          0.2510E+00-.5000E+00-.5000E+00
A00 4              10.1000E+000.4000E+00          0.3510E+00-.5000E+00-.5000E+00
A00 5              10.1000E+000.4000E+00          0.4510E+00-.5000E+00-.5000E+00
A00 6              10.1000E+000.4000E+00          0.5510E+00-.5000E+00-.5000E+00
A00 7              10.1000E+000.4000E+00          0.6510E+00-.5000E+00-.5000E+00
A00 8              10.1000E+000.4000E+00          0.7510E+00-.5000E+00-.5000E+00
A00 9              10.1000E+000.4000E+00          0.8510E+00-.5000E+00-.5000E+00
A0010              10.1000E+000.4000E+00          0.9510E+00-.5000E+00-.5000E+00
ina                                                                            I
A00 0              10.1000E-020.4000E-02          0.5000E-03-.5000E+00-.5000E+00 I

CONNE
A00 0A00 1                   10.5000E-030.5000E-010.1000E+01
A00 1A00 2                   10.5000E-010.5000E-010.1000E+01
A00 2A00 3                   10.5000E-010.5000E-010.1000E+01
A00 3A00 4                   10.5000E-010.5000E-010.1000E+01
A00 4A00 5                   10.5000E-010.5000E-010.1000E+01
A00 5A00 6                   10.5000E-010.5000E-010.1000E+01
A00 6A00 7                   10.5000E-010.5000E-010.1000E+01
A00 7A00 8                   10.5000E-010.5000E-010.1000E+01
A00 8A00 9                   10.5000E-010.5000E-010.1000E+01
A00 9A0010                   10.5000E-010.5000E-010.1000E+01
A0010A0011                   10.5000E-010.5000E-030.1000E+01

RPCAP----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
    9           .120       .02        3.
    7        0.45000   1.10e-1   8.0E-05      1.E6     1.0e0
COFT -----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
A00 0A00 1


GENER


INCON----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
A00 0           0.30000000E+00  Aqu
             2.700e6              00.0e0                1.00


ENDCY----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
```

Figure 12.7. Input file for example problem Test_1Pk (in Section 12.4) involving depressurization, equilibrium dissociation, and no inhibitor.

12.5. Problem Test_1Pk: Depressurization, Kinetic Dissociation, No Inhibitor

Hydrate dissociation is treated as a kinetic reaction in **Test_1Pk**. In all other respects, the problem is identical to that in Test_1P, as inspection of the input file in the directory "TestRuns" readily reveals. The output files corresponding to this simulation(**Test_1Pk.out**, **Hydrate_Info_1Pk**, **Conx_Time_Series_1Pk**) can also be found in the same directory.

Review of the output files reveals a dissociation patterns with pressure, temperature and saturation distributions entirely analogous to those for the equilibrium dissociation problem. As in the case of thermal stimulation (see Section 12.3), kinetic hydrate dissociation results in substantially longer execution times and slower time advance in the simulations. This is caused by the more computationally intensive system, the larger (because of the additional degree of freedom) and more difficult-to-solve matrix, and by the parameters of kinetic dissociation that impose time step limitations. These results are more thoroughly discussed in Section 12.6.

12.6. Problem Test_1P_Ice: Thermal Stimulation, Kinetic Dissociation, No Inhibitor, Ice Evolution

The only difference between this problem and that in **Test_1P** is in the boundary pressure at $x = 0$. By setting $P_b = 5\text{x}10^5$ Pa, the boundary is kept at a pressure significantly lower than that at the quadruple point of CH_4-hydrate. The lower pressure is expected to lead to faster and larger CH_4 release because of the significantly larger pressure differential that drives dissociation, as well as to the emergence of ice as the system tends to the boundary pressure. Of particular interest are the consequences of ice on the relative permeability of gas. Unless properly managed, the emergence of ice can

significantly reduce the pore space available to gas flow because ice has a lower density than the liquid water it replaces, and can adversely affect flow and gas production.

All the input and output files corresponding to the problem **`Test_1P_ice`** can be found in the directory `"TestRuns"`, where the standard TOUGH+ output **`Test_1P_ice.out`** (showing the changes in the phase regimes during the simulation) is also located. From its original state on the L+H phase regime, the rapid depressurization induces CH_4-hydrate dissociation, gas emergence, and the appearance of the L+G+H regime. As depressurization and dissociation continue, ice evolves as the system reaches the quadruple point, where it remains at a constant pressure and temperature until the hydrate is exhausted. Afterwards, the A+I+G phase regime appears, during which pressure changes but the temperature remains constant because of the liquid water-ice coexistence, and equal to that of the triple point of water. Because the x=0 boundary is kept at a temperature above freezing, ice melts eventually, and an aqueous and gas system remains.

The results of the illustrative problems in problems **`Test_1P`**, **`Test_1Pk`** and **`Test_1P_ice`** are compared in **Figures 12.8** to **12.11**. It should be noted that relatively coarse discretization has an effect on the results, but appears less pronounced than in the case of thermal stimulation.

Figure 12.8 shows the cumulative rates of CH_4 release from the dissociation of hydrate. Unlike the case of thermal dissociation, both kinetic and heat transfer limitations can affect dissociation in depressurization-induced gas release from hydrates. This is demonstrated in **Figure 12.8**, which shows very different initial rate patterns for the equilibrium and the kinetic problems. After the initial a burst of very rapid gas release at

the very beginning of the population (corresponding to the maximum deviation from equilibrium), the rate for kinetic dissociation at early times is substantially lower than that for equilibrium dissociation. The relationship is inversed at later times, and eventually the CH_4 release rates for kinetic and equilibrium dissociation become about the same as the entire system approaches equilibrium. As expected, the release rate of CH_4 from hydrate dissociation is substantially higher in problem **`Test_1P_ice`** because of the larger pressure differential. The higher dissociation rate leads to exhaustion of the hydrate at a much earlier time than in the other two problems, leading to zero release and denoted by the vertical segment of the curve at about t = 0.01 days.

In **Figure 12.9**, the cumulative gas volumes released from the hydrate in problems **`Test_1P`** and **`Test_1Pk`** are different at very early times, but become similar later. The cumulative gas rate for problem **`Test_1P_ice`** offers stark differences, and shows much larger volumes and exhaustion of the hydrate (marked by the flat part of the curve). The obvious implication is that, as long as potentially adverse relative permeability effects can be avoided, enhanced depressurization provides faster rates and early gas recovery. Compared to the **Figures 12.3** and **12.4**, **Figures 12.5** and **12.6** convincingly demonstrate the superiority and efficiency of depressurization as a method of gas production from gas hydrates. Note that **Figures 12.8 and 12.9** were plotted using the data from the output files **`Hydrate_Info_XXX`**.

In these problems, gas is produced as it escapes into the low-pressure boundary at x = 0. **Figures 12.10** and **12.11** (based on data from the **`Conx_Time_Series_XXX`** output files) show the volumetric production rate of CH_4 and the cumulative CH_4 production. Because of the L+H initial regime, all the CH_4 originates from the dissociation of the

hydrate. **Figure 12.10** shows that the CH_4 production rate (i.e., the fluxes across the x = 0 boundary) for kinetic dissociation lags behind that for equilibrium dissociation at a very early time, then exceeds it, and finally the two become roughly equal at later times. The production rate for the **`Test_1P_ice`** problem is significantly larger, and becomes zero at a very early time because no hydrate is left in the system (see the **`Conx_Time_Series_1P_ice`** file in Appendix A3).

In **Figure 12.11**, the total gas production (measured as the CH_4 volume that crosses the $x = 0$ boundary) from the kinetic dissociation problem remains consistently lower than that for the equilibrium dissociation case, and the maximum deviation occurs at a very early time. Similarly to the pattern observed in **Figure 12.9**, gas production in the **`Test_1P_ice`** problem is significantly larger and leads to very early disappearance of the hydrate. A comparison of Figures **12.10** and **12.11** to the analogous ones for thermal stimulation (**Figures 12.5** and **12.6**) further confirm the superiority of depressurization as a dissociation strategy.

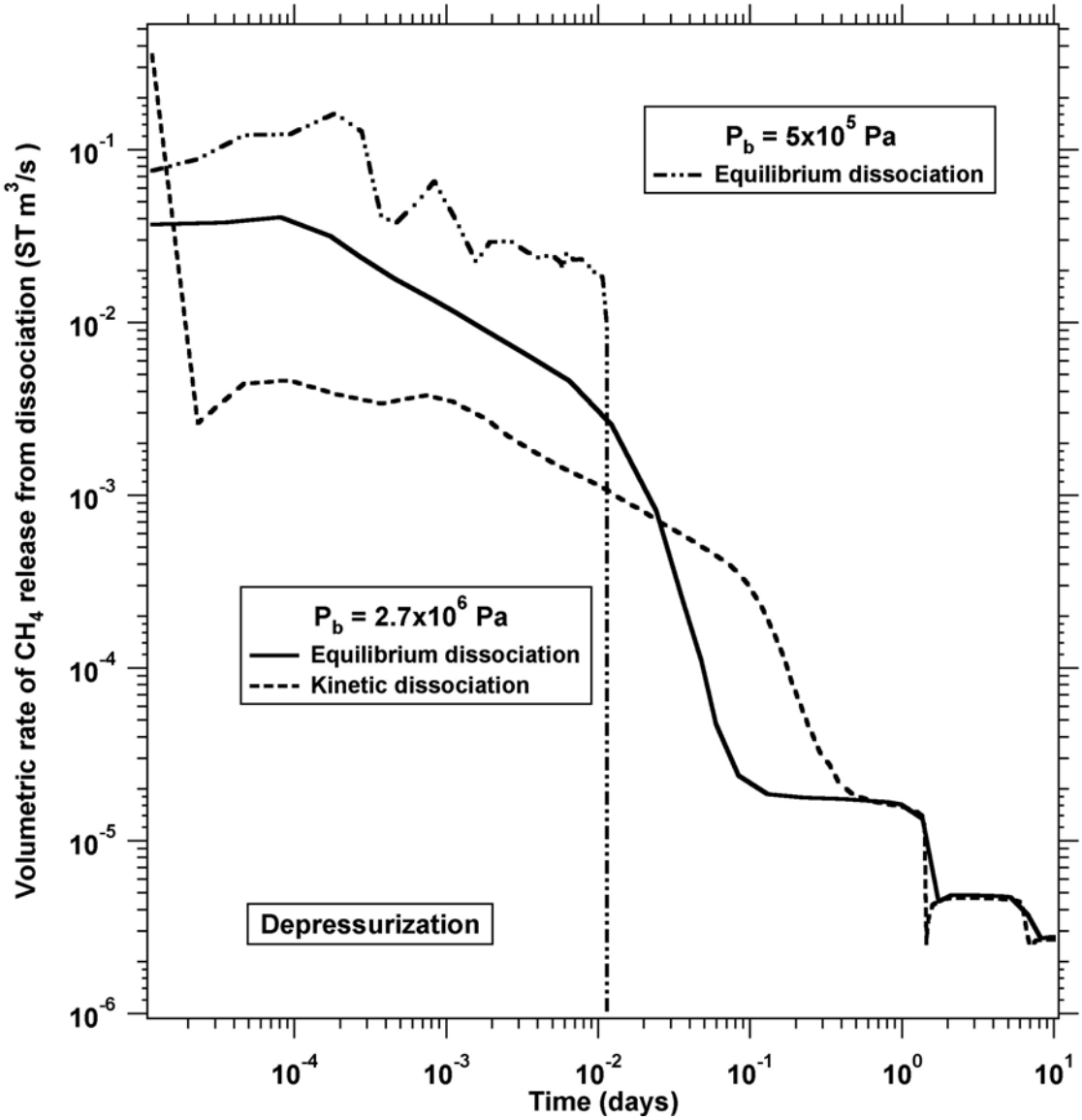

Figure 12.8. Comparison of the volumetric rates of CH_4 release from depressurization-induced hydrate dissociation in problems `Test_1P, Test_1Pk` and `Test_1P_ice`.

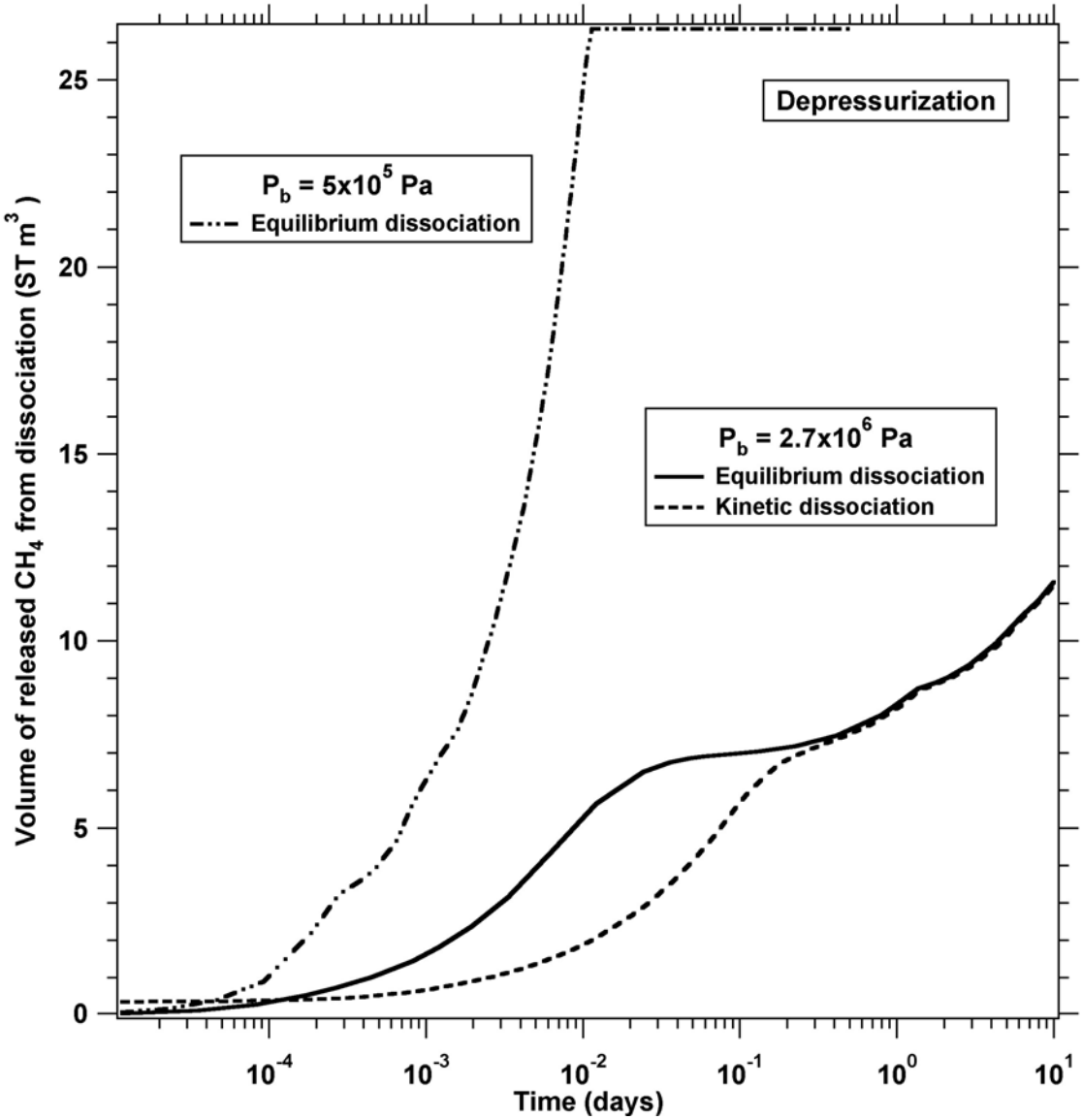

Figure 12.9. Comparison of the cumulative volumes of CH_4 released from depressurization-induced hydrate dissociation in problems `Test_1P, Test_1Pk` and `Test_1P_ice`.

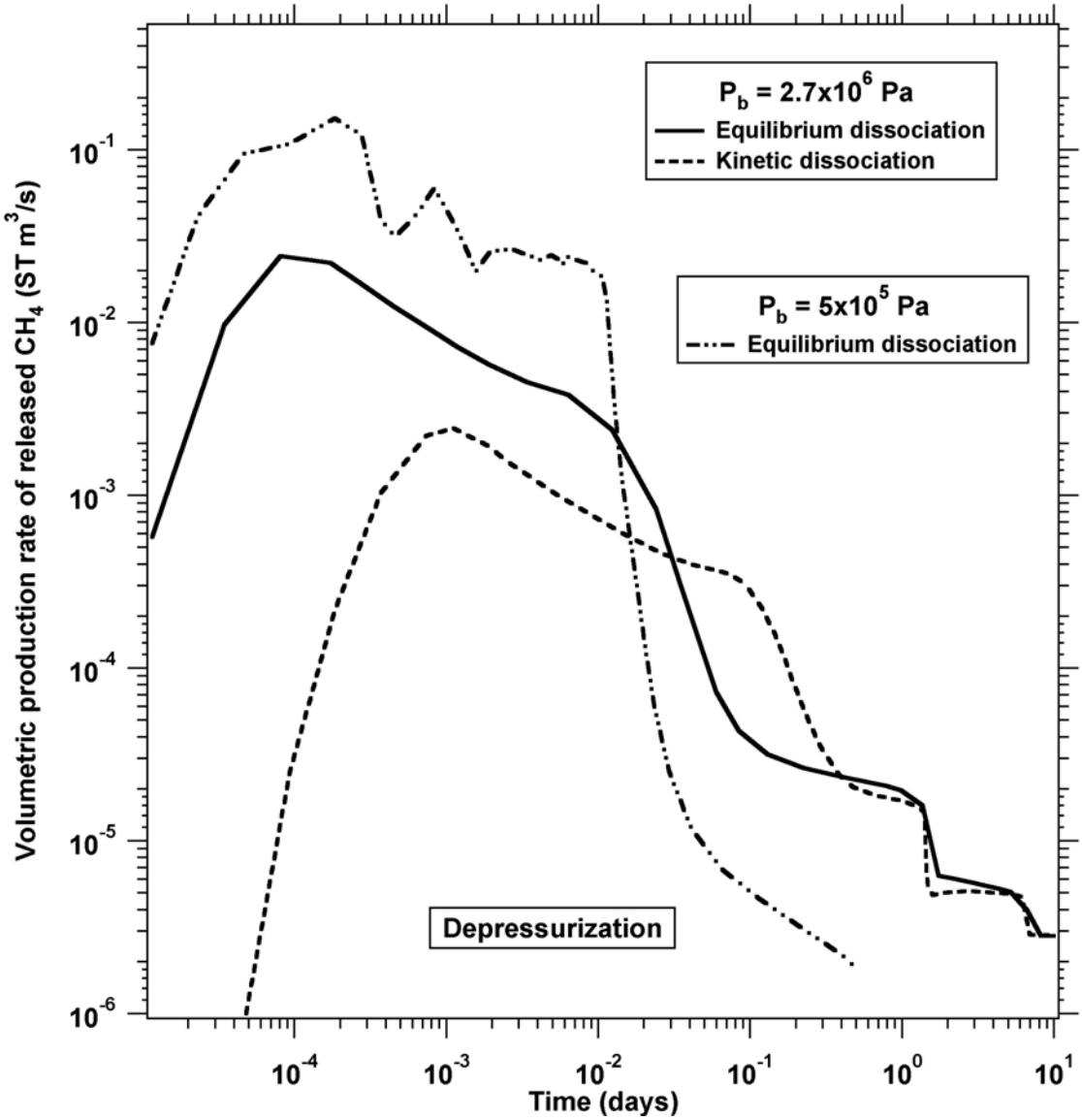

Figure 12.10. Comparison of the volumetric production rates of CH_4 (measured as the flux crossing the x = 0 boundary) in problems `Test_1P`, `Test_1Pk` and `Test_1P_ice`.

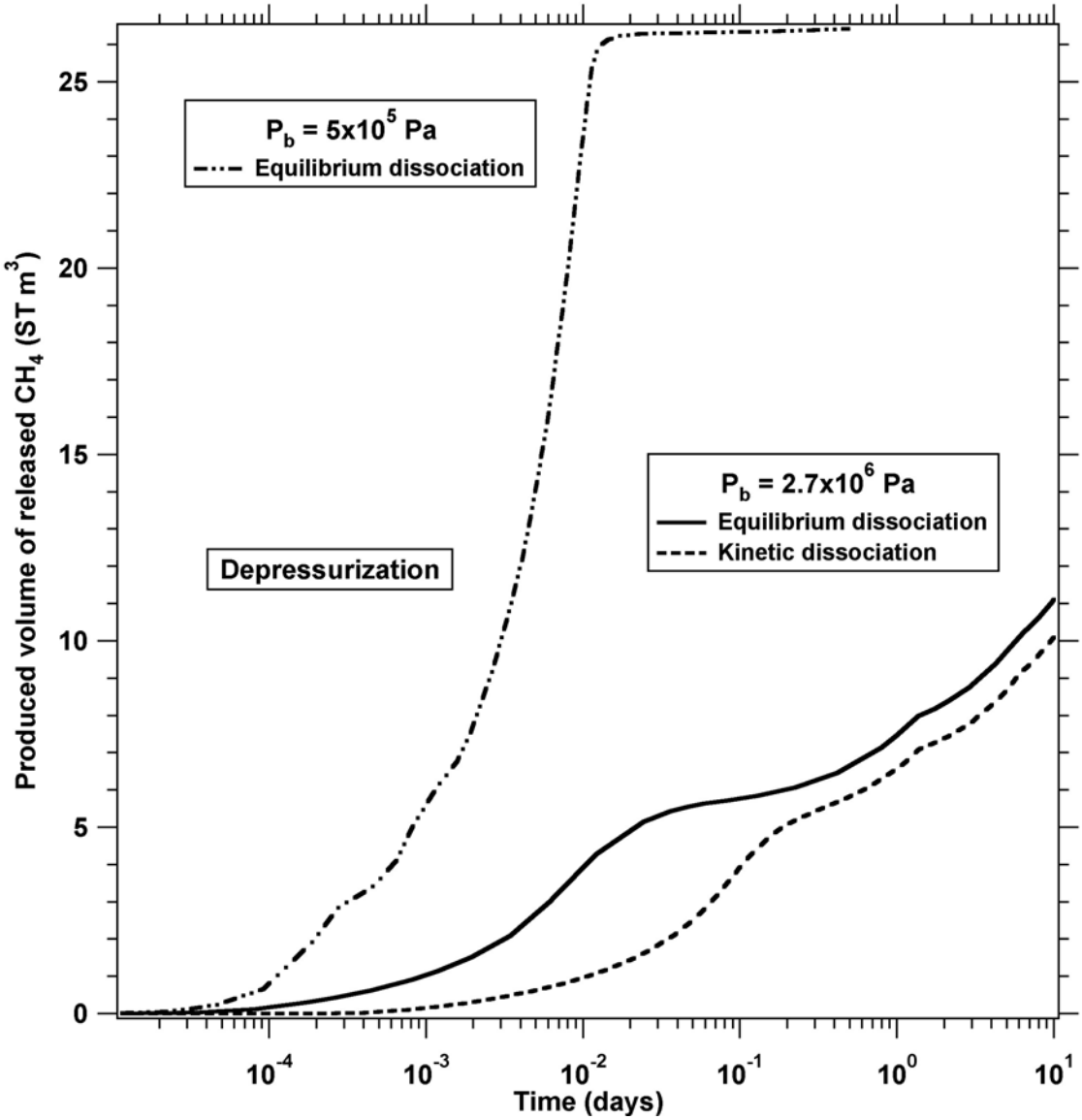

Figure 12.11. Comparison of the total production volumes of CH_4 (measured as the total volumes crossing the x = 0 boundary) in problems `Test_1P`, `Test_1Pk` and `Test_1P_ice`.

12.7. Problem Test_1TbS and Test_1TbSk: Hydrate Formation, Equilibrium and Kinetic Hydration reaction, Inhibitor

The porous medium in 1-D columns in problems **`Test_1TbS`** and **`Test_1TbSk`** has the same properties as those reported in problem **`Test_1P`** (see Section 12.2), but the systems have different initial and boundary conditions. Because the pressure P = 4x10^6 Pa and temperature T = 8.5 ^{o}C are outside the stability zone of hydrate, the pore space is filled by a two-phase system involving a gas and an aqueous phase. The aqueous phase is a NaCl solution, in which the initial mass fraction of the salt is $X = 10^{-3}$. The column is attached to a constant pressure boundary at $x = L_{max}$ that can provide fluids to the active domain. The x = 0 boundary is impermeable, thermally conductive, and at a constant temperature of T_b = 1.2 ^{o}C. Because of conduction, the temperature in the column is expected to drop below the hydration temperature at the prevailing pressure, leading to the formation of hydrate. The temperature decline is somewhat buffered by the exothermic reaction of hydrate formation, and the formation process is affected by the presence of an inhibitor. In addition to the original equilibrium problem, a second case was investigated, in which the initial temperature was higher, with T = 4.5 ^{o}C. The lower initial T is expected to lead to earlier and faster hydrate formation.

An equilibrium hydrate reaction is assumed in problem **`Test_1TbS`**, while a kinetic hydrate reaction is assumed in **`Test_1TbSk`**. The input files and all the corresponding output files are available in the directory `"TestRuns"`. The outputs corresponding to the lower initial T have the identifier '**`_1TbS2`**'. A review of the standard TOUGH+ output files clearly shows the evolution of hydrate. Of particular interest is the increase

in the inhibitor concentration as hydrate is formed. This is consistent with expectations because inhibitors cannot be partitioned into the solid hydrates. As in all previous kinetic cases, the kinetic treatment of the hydration equation results in longer execution times and slower advancement of the simulation time.

Figure 12.12 shows the cumulative rates of CH_4 consumption during hydrate formation. With the exception of results at very early times (when the rate corresponding to hydrate formation is larger because of maximum deviation from equilibrium), the CH_4 consumption rate for $T = 8.5\ ^{o}C$ are practically identical for both kinetic and equilibrium hydrate reaction. This was expected because the heat removal occurs through conduction, which is a very slow process and dominates the reaction. Additionally, the onset of hydrate evolution occurs at the same time for both cases.

The CH_4 consumption rate for the $T = 4.5\ ^{o}C$ equilibrium reaction case is significantly larger, and hydrate evolves at an earlier time. These results are reflected in **Figure 12.13**, which shows the cumulative CH_4 consumption during the formation process. The total volume of reacted CH_4 is the same for both the $T = 8.5\ ^{o}C$ kinetic and equilibrium reaction, while it is much larger for the $T = 4.5\ ^{o}C$ equilibrium reaction (i.e., the amount consumed is much lower).

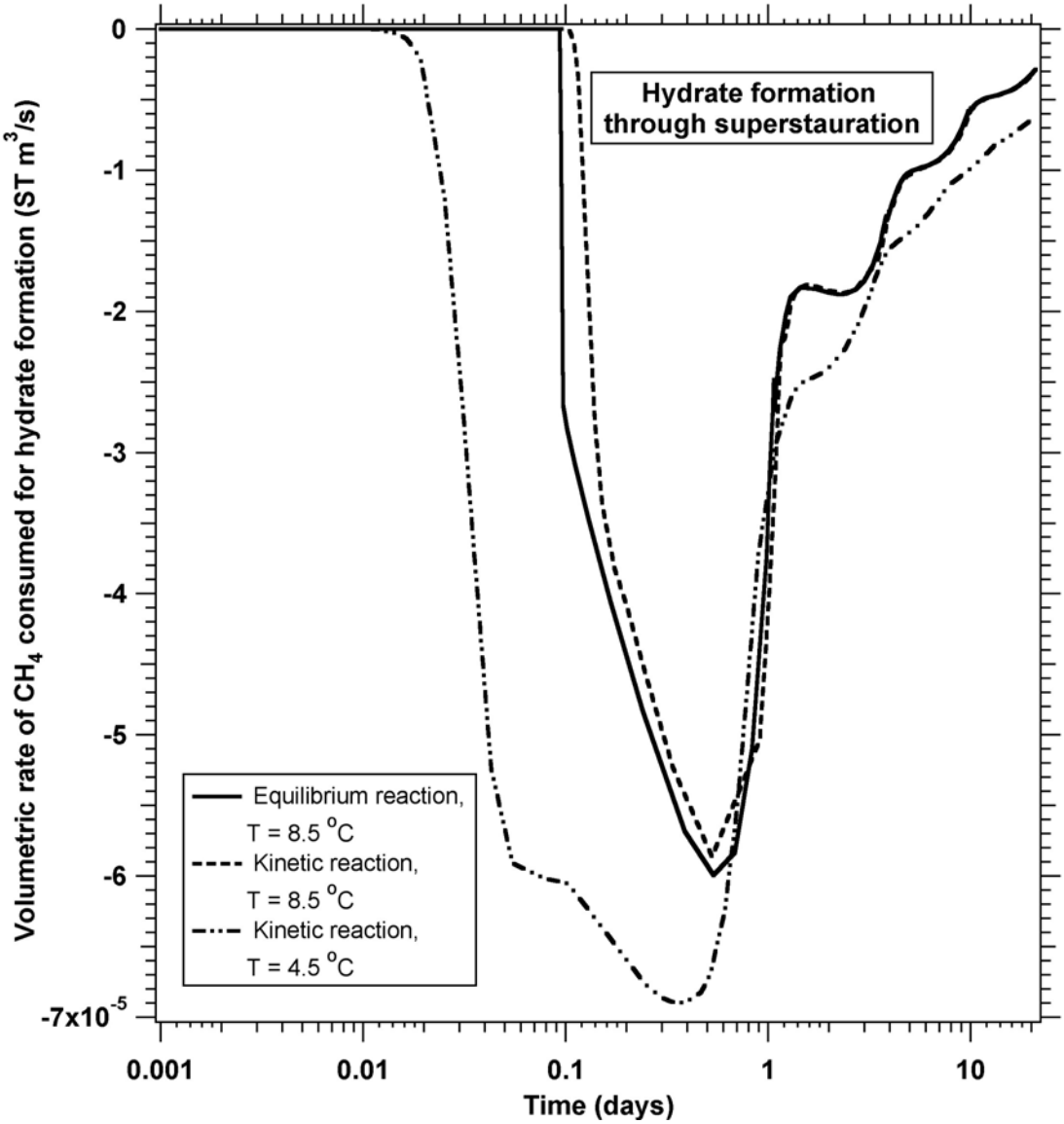

Figure 12.12. Comparison of the volumetric rates of CH_4 consumption during hydrate formation in problems `Test_1TbS, Test_1TbSk` and `Test_1TbS2`.

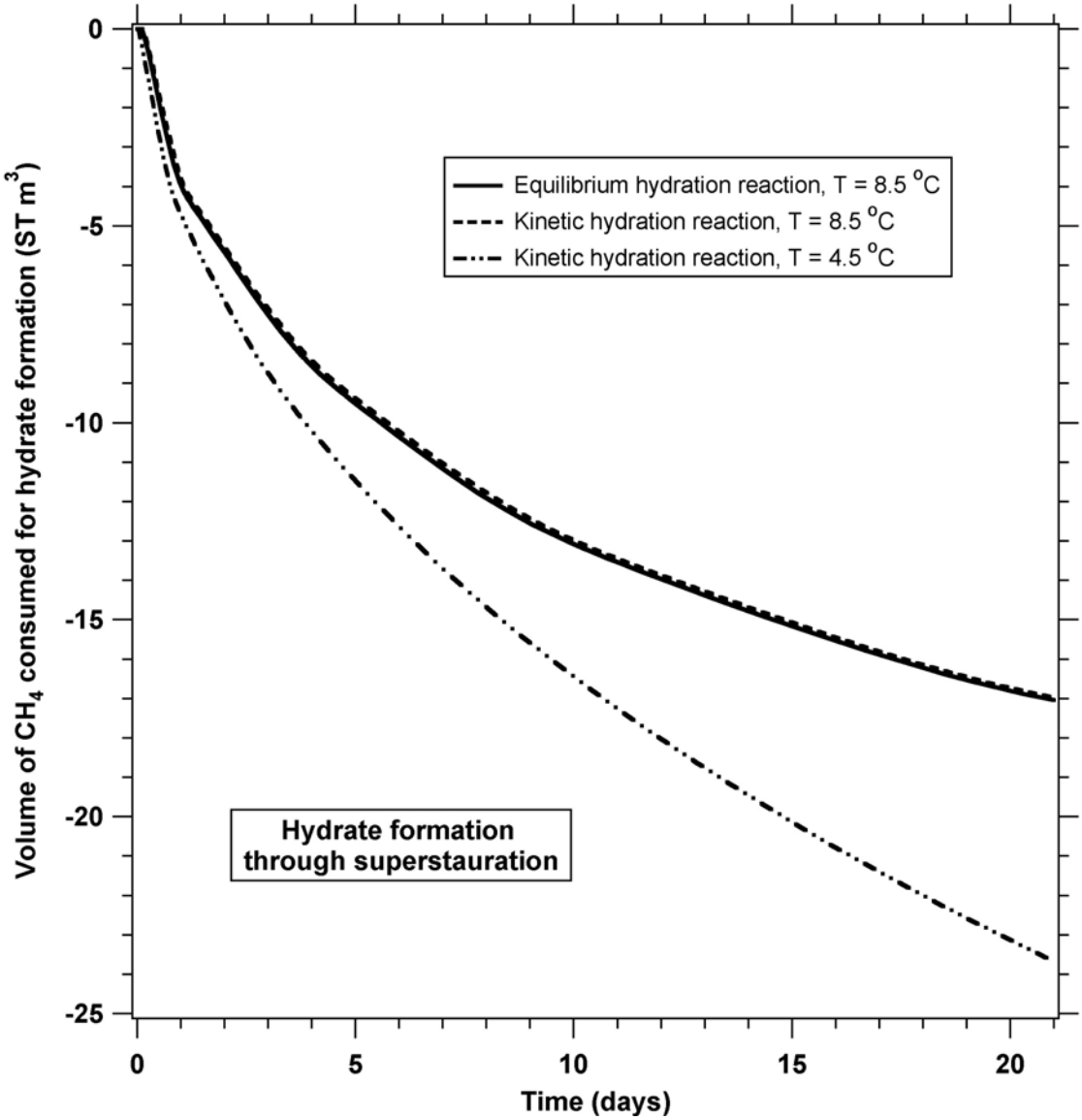

Figure 12.13. Comparison of the cumulative volumes of CH_4 consumed during hydrate formation in problems `Test_1TbS, Test_1TbSk` and `Test_1TbS2`.

12.8. Problem Test_2Qp: Equilibrium Hydrate Dissociation, Depressurization, Radial Grid - Single Well

Problem **`Test_2Qp`** represents a more realistic scenario of gas production from a Class 3 (Moridis and Collett, 2004) hydrate deposit in which water and hydrate are initially at equilibrium. In the deposit, P = 9.8x10^6 Pa, T = 12.5 ^{o}C, and the initial hydrate saturation is S_H = 0.5. Such conditions can be encountered at the bottom of the stability zone in permafrost accumulations (Moridis et al., 2003). The input file and all the corresponding output files are available in the directory `"TestRuns"`.

This problem involves production from a 10-m thick cylindrical reservoir that extends to R_{max} = 224 m. The properties of the formation are the same with those in all previous problems (see Section 12.1). A Dirichlet-type (constant conditions equal to the initial conditions) boundary is assumed at R_{max}, i.e., the outer rim of the grid. Fluids are withdrawn at a mass flow rate of Q = 1 kg/s through a well at the center of the grid, and are distributed in the production stream according to their mobilities. The fluid withdrawal causes a pressure decline that leads to the depressurization-induced release of CH_4.

The grid in the problem is much larger than in the previous examples, but still insufficient to provide the resolution needed for the representation of an infinite-acting system. Because of this, the proximity of the boundary to the well, the near incompressibility of the aqueous phase, and the quantum-type system behavior in equilibrium dissociation, discretization effects are expected to appear in the estimation of the dissociation rates (see earlier discussions).

The evolution of (a) the cumulative rate of CH_4 release from dissociation and (b) the cumulative volume of CH4 released from dissociation in the repository are shown in **Figures 12.14** and **12.15**, respectively. As expected, the curve of the cumulative rate of CH4 release exhibits early oscillations attributed to both the rather coarse discretization and the quantum nature of hydrate dissociation. As expected, the release rate increases initially because of advancing depressurization, but begins to decline after a maximum is reached. This decline is caused by the rapid cooling of the hydrate because of advancing dissociation, which affects the rate of dissociation (and the consequent CH_4 release) of the remaining hydrate. As discussed earlier, the jagged appearance of the rate curve in **Figure 12.14** is caused by the coarse discretization. Conversely, the curve of the cumulative volume of released CH_4 is smooth, with the maximum release rate marked by the inflection point at about t = 16 days.

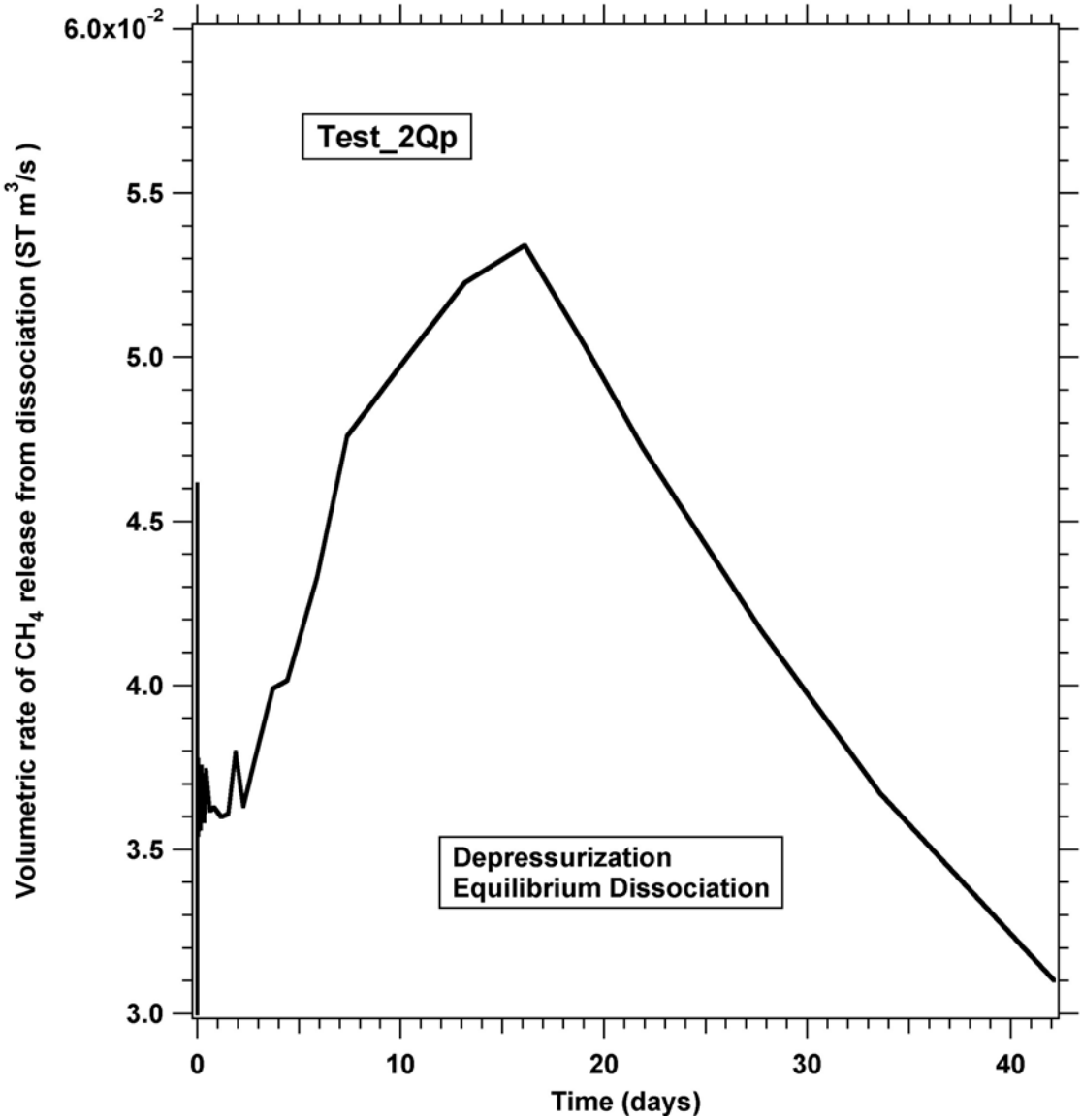

Figure 12.14. Volumetric rate of CH_4 release during hydrate dissociation in problem `Test_2Qp`.

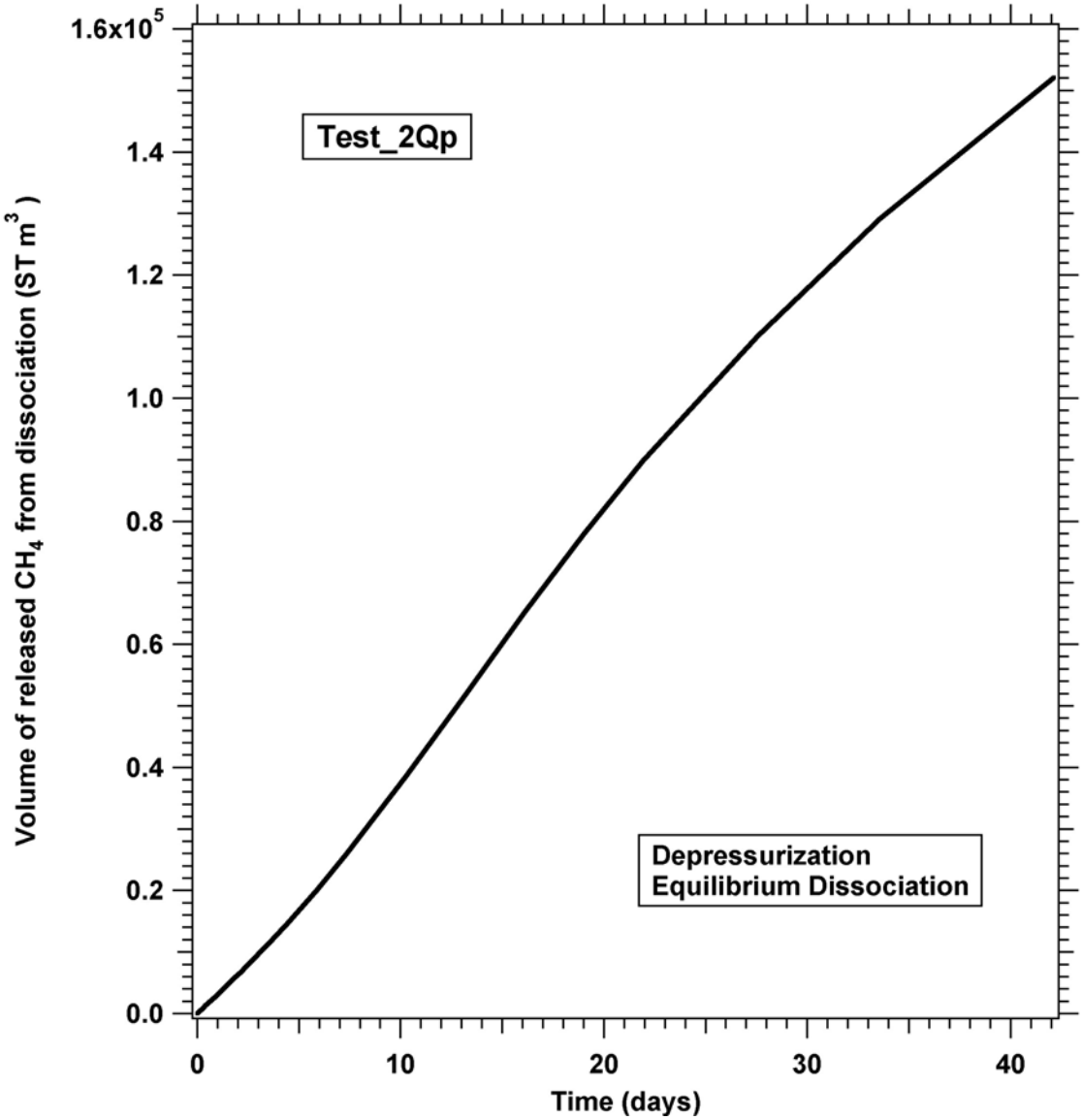

Figure 12.15. Cumulative volume of CH_4 released during hydrate dissociation in problem `Test_2Qp`.

12.9. Problem Test_3Qpk: Kinetic Hydrate Dissociation, Depressurization, Radial Grid - Single Well with Wellbore Heating

Problem **`Test_3Qpk`** uses the same grid, and porous medium and boundary conditions as problem **`Test_2Qp`**. This deposit, for which P = 9.51x10^6 Pa and T = 12.5 ^oC, involves a deposit in which water, gas and hydrate are initially at equilibrium. The initial gas and hydrate saturations are S_G = 0.5 and S_H = 0.4, respectively. A Dirichlet-type boundary (constant conditions equal to the initial conditions) is assumed at R_{max}, i.e., the outer rim of the grid. Fluids are withdrawn at a mass flow rate of Q = 0.3 kg/s through a well at the center of the grid, and are distributed in the production stream according to their mobilities. The fluid withdrawal causes a pressure decline that leads to the depressurization-induced release of CH_4. To alleviate potential hydrate buildup problems that can severely restrict flow to the well (possibly because of temperature drop and availability of gas and water in the vicinity of the well), the wellbore is heated at a rate of Q_H = 1.2 KW. The input file and all the corresponding output files are available in the directory `"TestRuns"`.

The evolution of the rates of (a) cumulative CH_4 release in the deposit, and (b) gas production at the well are shown in **Figure 12.16**. Because of the coexistence of the three phases in the initial system, the curve of the CH_4 release rate is smooth. A comparison of these two curves reveals that the contribution of gas from hydrate dissociation to the rate of gas production decreases progressively with time, and becomes minor at the end of the simulation period. The reason for this performance is the continuing cooling of the hydrate as it dissociates, in addition to the proximity of the well

to the (constant-conditions) boundary and the relatively large permeability, which do not allow efficient depressurization of the system. Consequently, the quasi steady-state condition that is reached rather early is not conducive to increased gas production because the system adjusts to a new state of equilibrium. The novice user is encouraged to investigate the system behavior if the Dirichlet-type boundary is replaced by a no-flow Neuman boundary.

The cumulative volumes of (a) CH4 released from dissociation in the repository and (b) produced from the wells are shown in **Figure 12.17**. The declining contribution of CH_4 from dissociation is reflected in the practically flat portion of the released CH_4 curve.

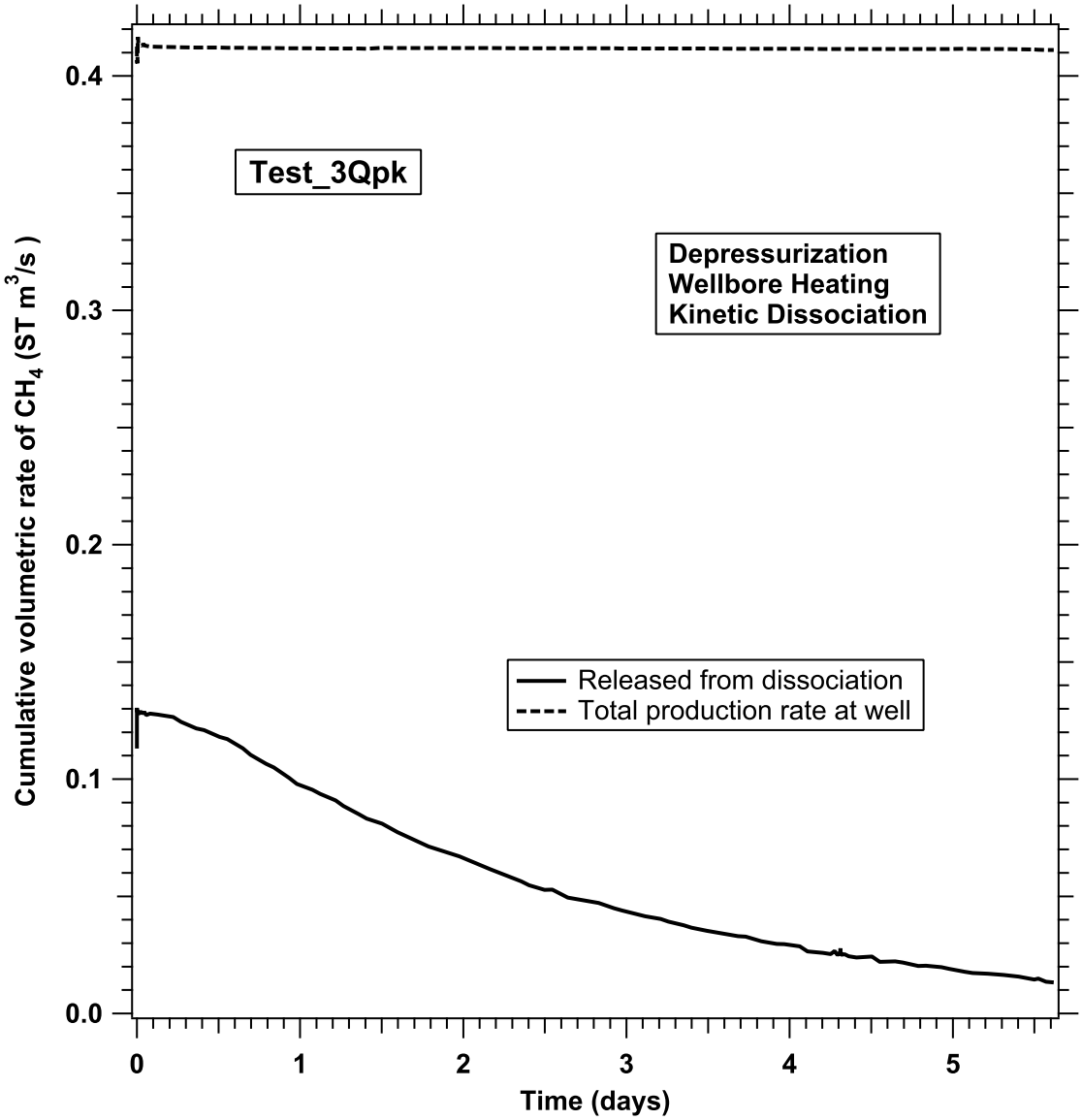

Figure 12.16. Volumetric rates of CH_4 (a) release in the reservoir during hydrate dissociation and (b) production from the well in problem `Test_3Qpk`.

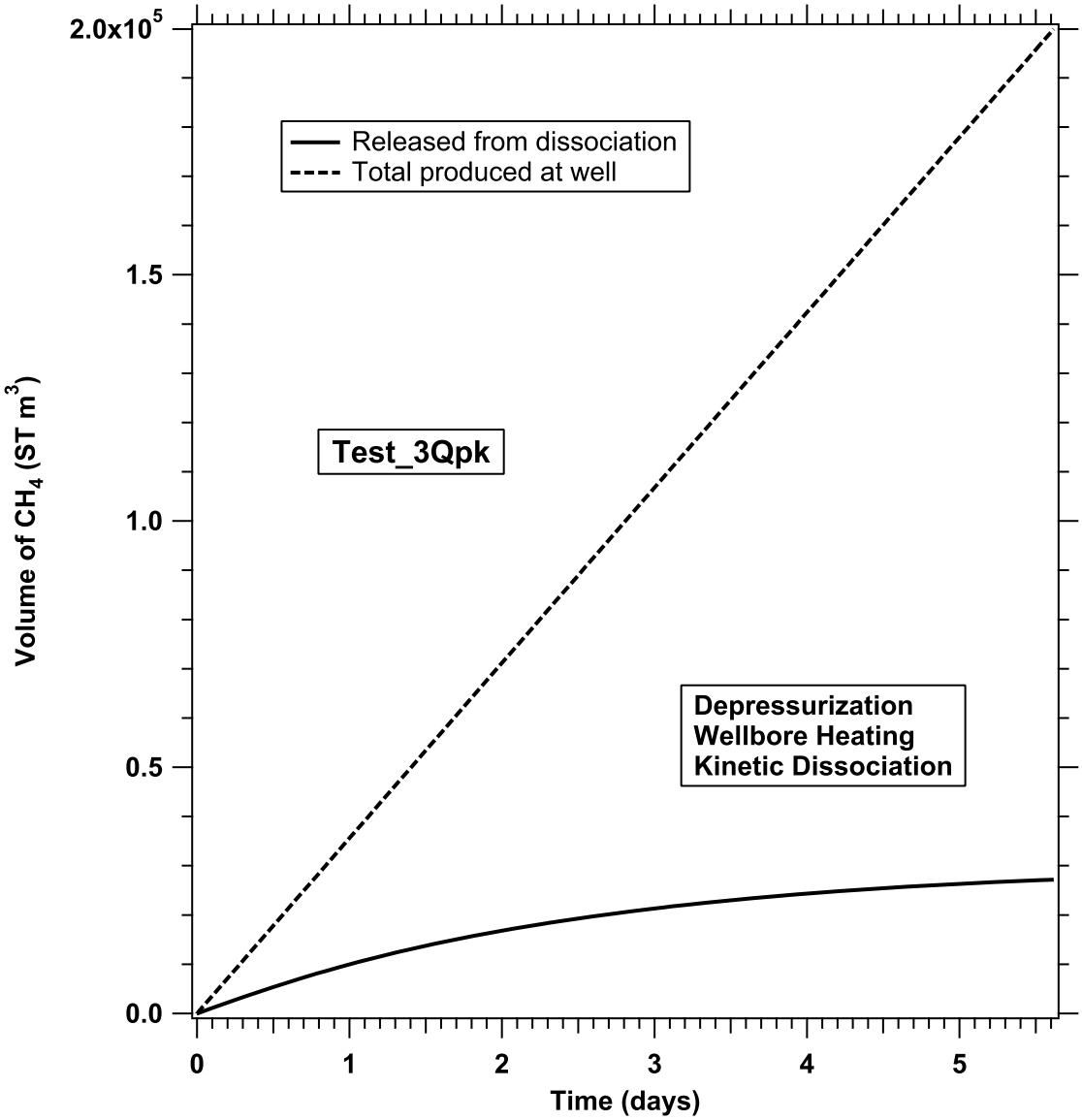

Figure 12.17. Cumulative volumes of CH_4 (a) released in the reservoir during hydrate dissociation and (b) produced from the well in problem `Test_3Qpk`.

12.10. Problem Test_2D: Equilibrium Hydrate Dissociation, 2-D Areal System

Problem **`Test_2D`** describes gas production from a realistic areal 2-D hydrate-bearing formation that has the same porous medium properties as all the previous examples. The simulation domain consists of a square system with a side of 50 m and a formation thickness of 10 m. The domain is subdivided in 50 x 50 = 2500 cells in (x,y), with the well located in the cell at (x,y) = (0.5 m, 0.5 m). This is considered a part of a large-scale production system that involves a deposit of considerable areal extent and a large number of producing wells. Under these conditions, the simulation domain represents one quarter of the basic production stencil, and its outer boundaries are no-flow Neuman-type boundaries because of symmetry. The 2-D geometry of the system automatically implies no flow in the third direction, i.e., the system is treated as insulated along its top and bottom boundaries. While this is not strictly true in porous media, such heat transfer through impermeable boundaries can only occur through conduction, which is a very slow process and is not expected to appreciably affect the results. Additionally, by ignoring heat contribution from its boundaries, this simulation provides the worst-case scenario of gas production from such a hydrate accumulation.

In this deposit, water, gas and hydrate are initially at equilibrium, and the pressure is the hydration pressure corresponding to T = 12.5 ^{o}C. The initial gas, aqueous and hydrate phase saturations are S_G = 0.1, S_A = 0.3 and S_H = 0.6, respectively. Fluids are withdrawn at a mass flow rate of Q = 0.5 kg/s through the production well, and are distributed in the production stream according to their mobilities. The fluid withdrawal causes a pressure decline that leads to the depressurization-induced release of CH_4. The production flow

rate remains constant, and is certain to lead to temperature decline and ice appearance because of the endothermic nature of dissociation. The input file and all the corresponding output files are available in the directory `"TestRuns"`. Because of the computationally intensive nature of this realistic problem, the simulation period was limited to two days. Interested users are encouraged to run this problem (modified per their desires) for longer periods.

Figure 12.18 shows the evolution of the cumulative volume of CH4 released from dissociation in the repository in the first two days of production. Note that the released volume increases practically linearly with time. In this short period, a daily release of about 10000 ST m^3 from the hydrate is observed. This is expected to change at later times, when exhaustion of the 'free' gas and hydrate resources in the reservoir will inevitably lead to a sublinear performance.

The pressure distribution in the reservoir at t = 2 days in **Figure 12.19** indicates substantial pressure declines over an extended portion of the domain, indicating significant hydrate dissociation (as supported by the results in **Figure 12.18**). The corresponding temperature distribution in **Figure 12.20** shows an extended cool region even at that early time, providing additional indirect evidence of dissociation.

Direct evidence of dissociation is provided by the corresponding hydrate distribution at the same time in **Figure 12.21**, which clearly indicates severe hydrate dissociation within 5 m from the well, and measurable dissociation as far as 15 m away from the well. While these are very positive and encouraging results, they are not indicative of the production potential if there is no information on the relative permeability regime in the vicinity of the wellbore. The production potential of the hydrate accumulation can be

seriously diminished if the water (released from the hydrate dissociation) or the ice (that can be formed in the course of the endothermic reaction) inhibit the flow of gas toward the well.

The answer to this question is provided by the distributions of the aqueous and gas phase saturations at t = 2 days in **Figures 12.22** and **12.23**. While the water saturation increases in the vicinity of the well in the process of dissociation, the gas saturation increases also. Such a desirable regime then becomes a production management and design issue in the design of appropriate production strategies from the hydrate deposit.

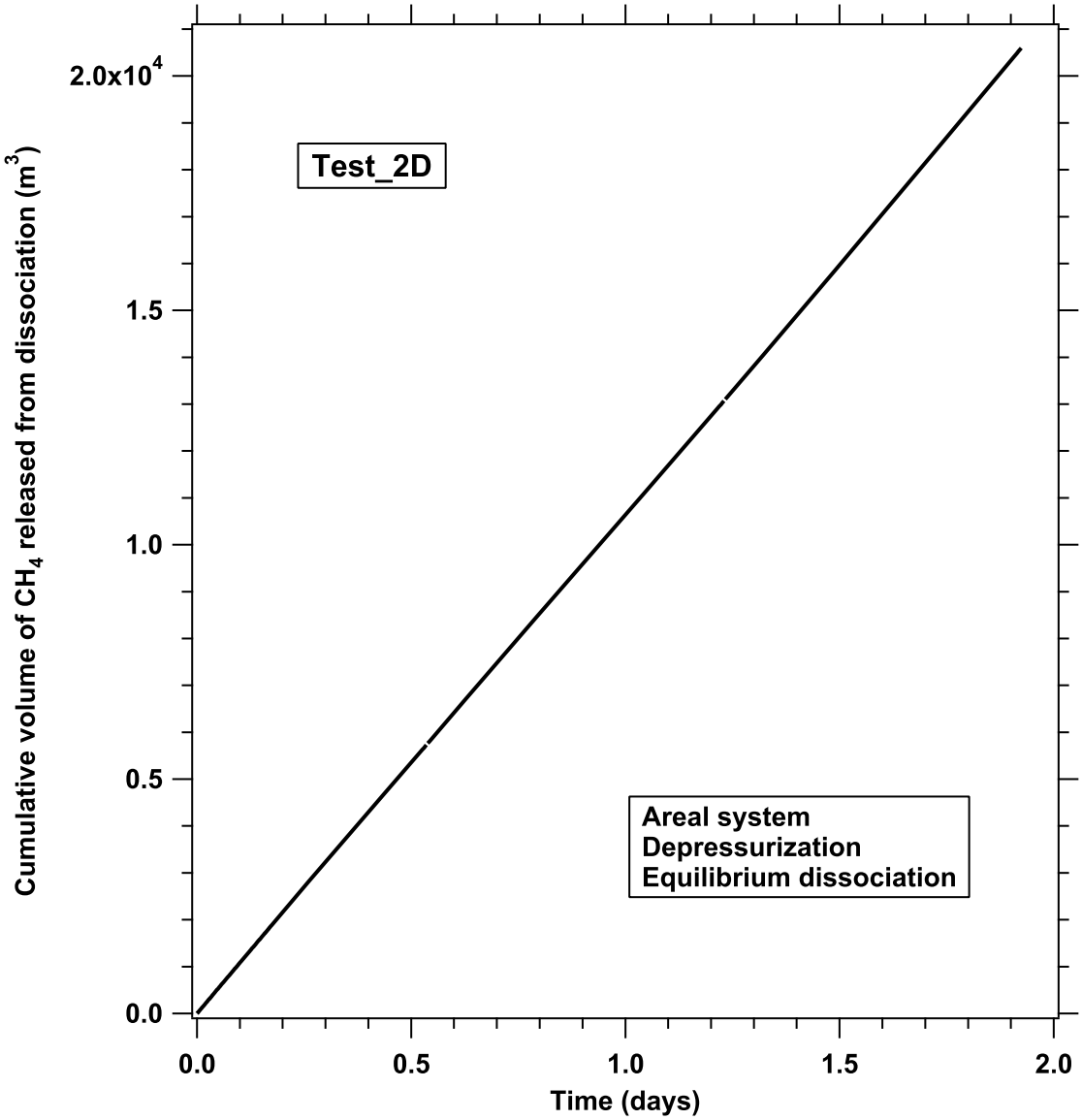

Figure 12.18. Volumetric rates of CH_4 release from the hydrate in the reservoir during gas production in problem `Test_2D`.

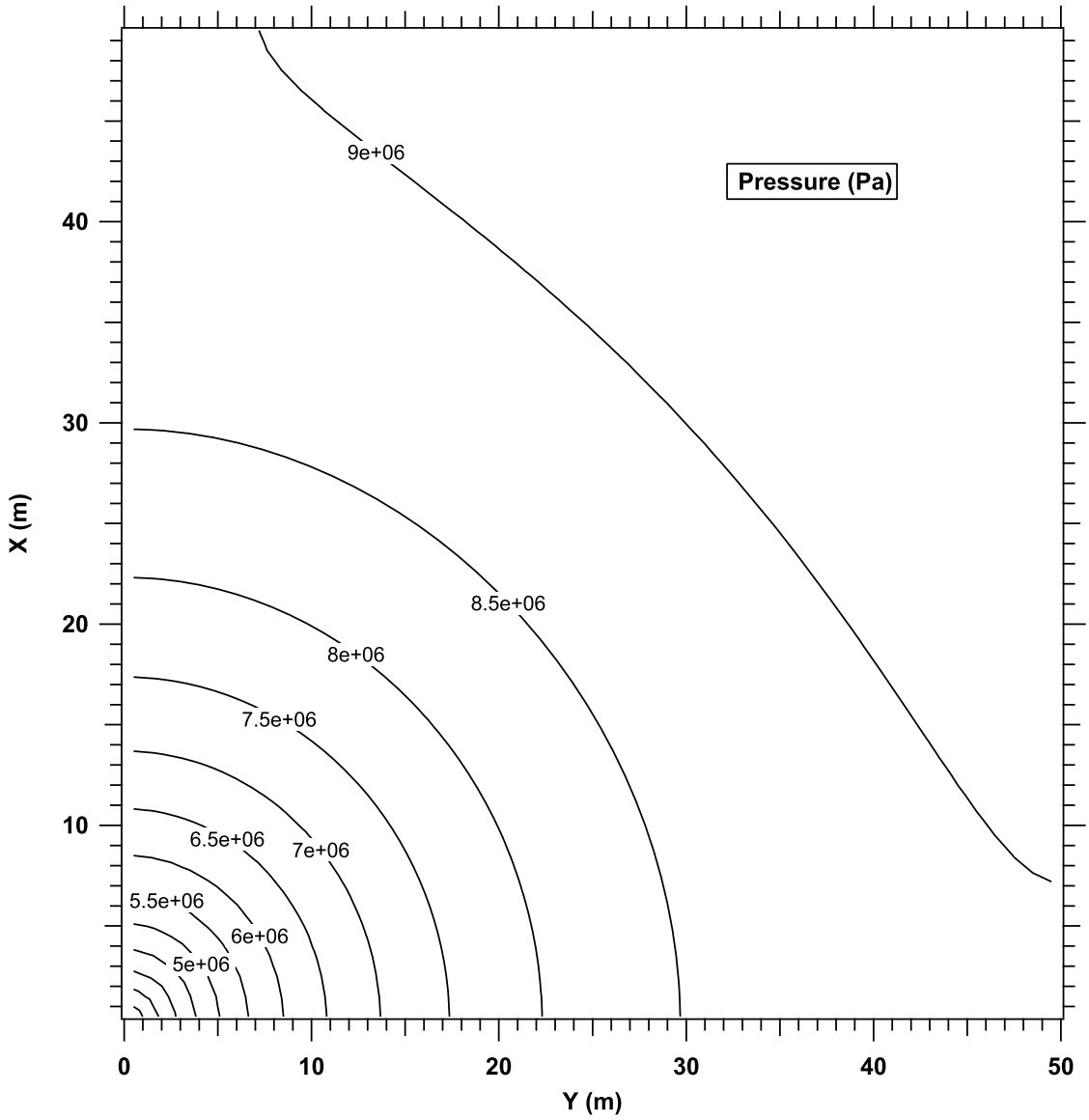

Figure 12.19. Pressure distribution in the reservoir at t = 2 days in problem `Test_2D`.

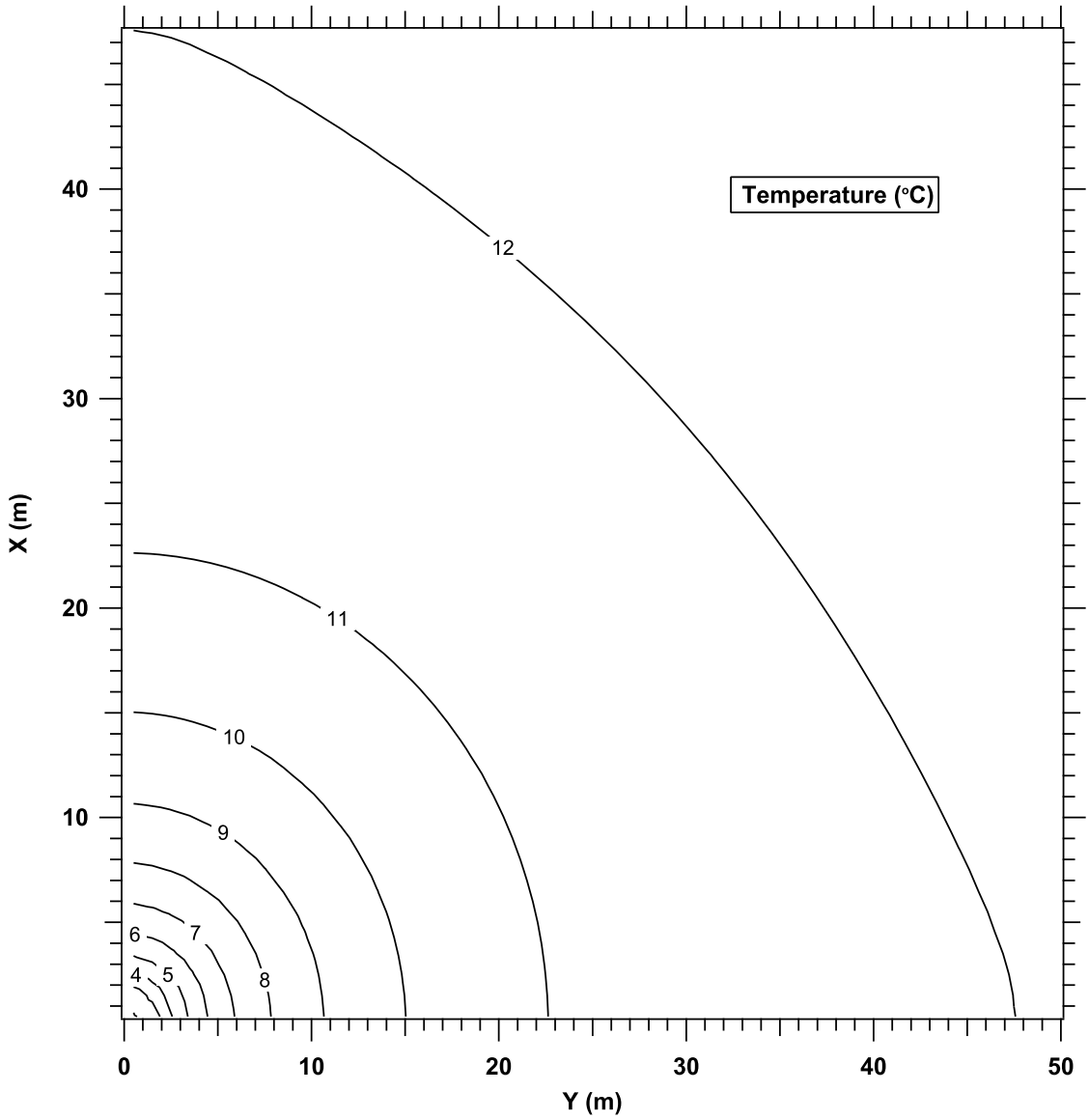

Figure 12.20. Temperature distribution in the reservoir at t = 2 days in problem `Test_2D`.

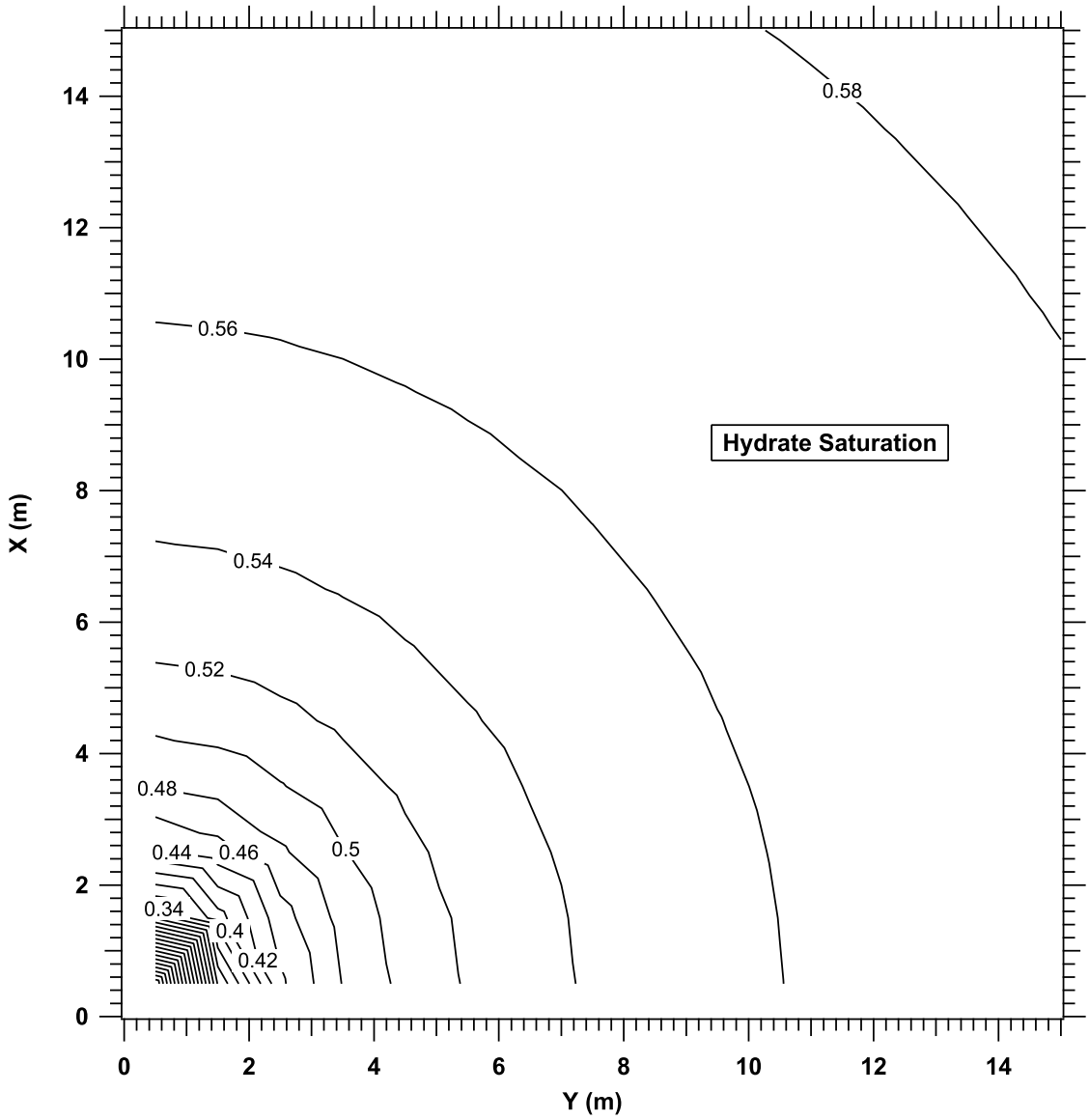

Figure 12.21. Hydrate saturation distribution in the reservoir at t = 2 days in problem `Test_2D`.

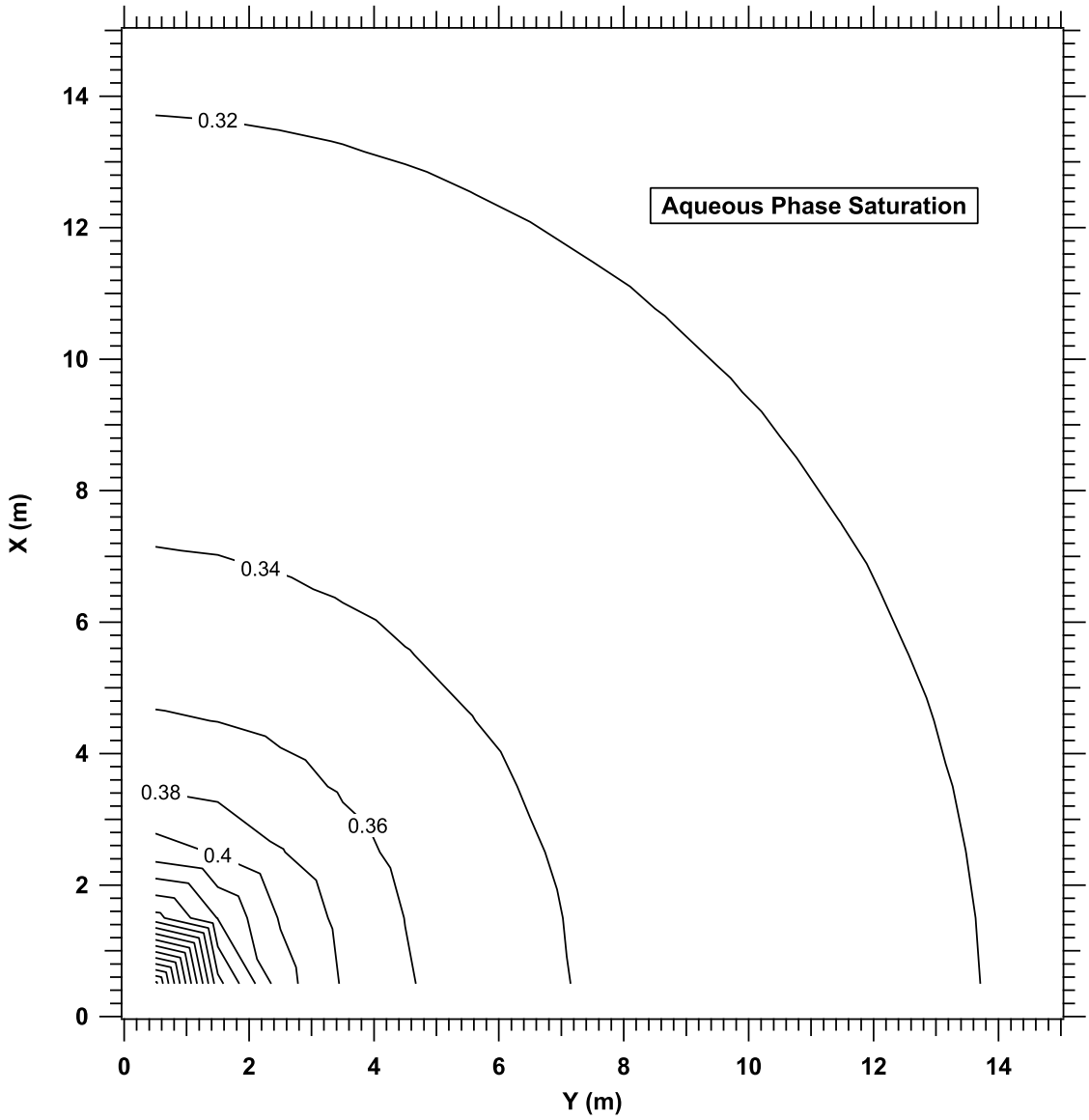

Figure 12.22. Aqueous phase saturation distribution in the reservoir at t = 2 days in problem `Test_2D`.

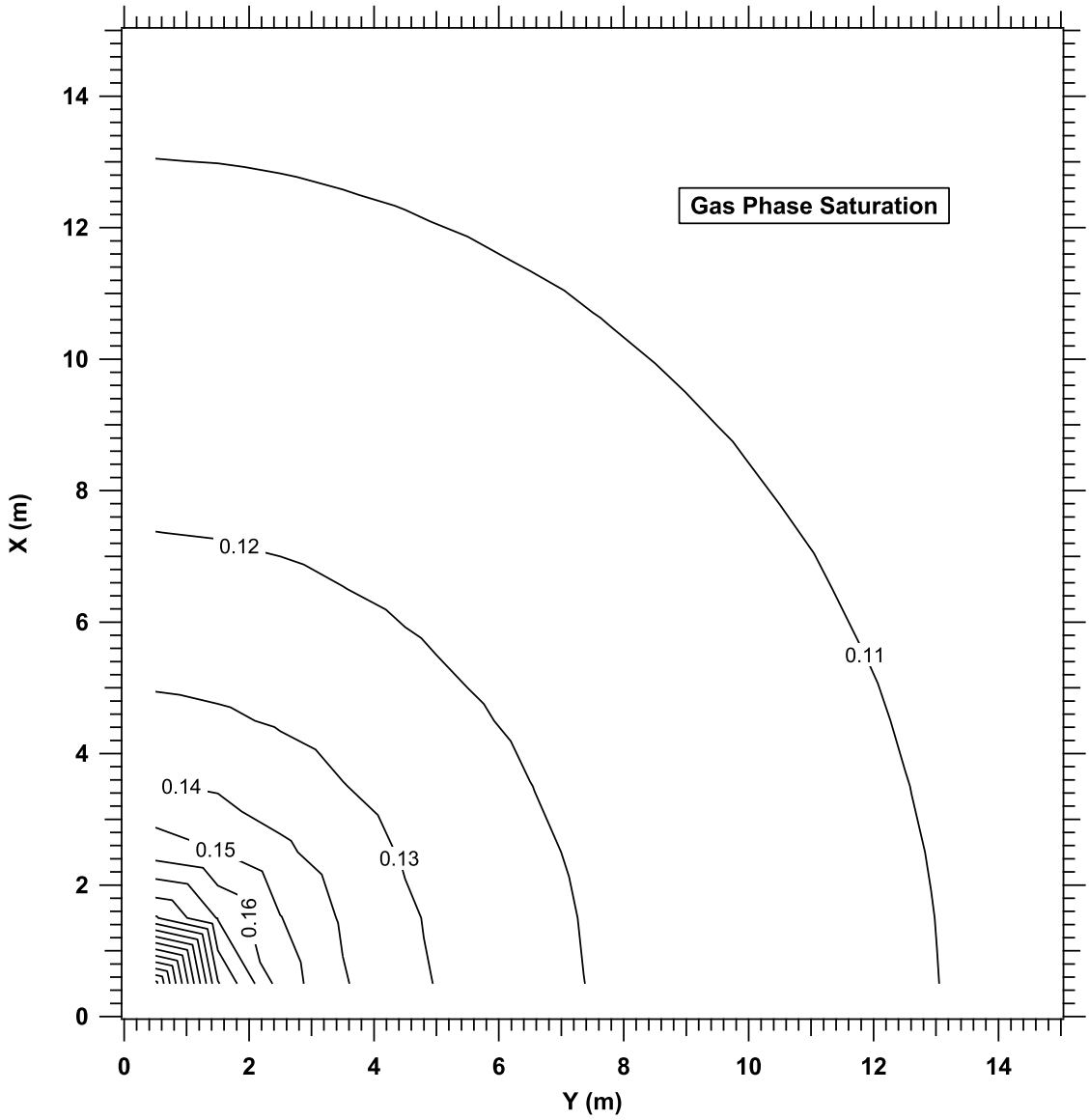

Figure 12.23. Gas saturation distribution in the reservoir at t = 2 days in problem `Test_2D`.

Acknowledgements

This work was supported by the Assistant Secretary for Fossil Energy, Office of Natural Gas and Petroleum Technology, through the National Energy Technology Laboratory, under the U.S. Department of Energy, Contract No. DE-AC03-76SF00098.

References

Atkinson, P.G., R. Celati, R. Corsi and F. Kucuk. Behavior of the Bagnore Steam/CO2 Geothermal Reservoir, Italy, *Soc. Pet. Eng. J.*, 228 - 238, 1980.

Aunzo, Z.P., G. Bjornsson and G.S. Bodvarsson. Wellbore Models GWELL, GWNACL, and HOLA, Lawrence Berkeley National Laboratory Report LBL-31428, Berkeley, CA, October 1991.

Aziz, K. and A. Settari. *Petroleum Reservoir Simulation*, Elsevier, London and New York, 1979.

Battistelli, A., C. Calore and K. Pruess. The Simulator TOUGH2/EWASG for Modeling Geothermal Reservoirs with Brines and Non-Condensible Gas, *Geothermics*, Vol. 26, No. 4, pp. 437 - 464, 1997.

Bird, R., W.E. Stewart and E.N. Lightfoot. *Transport Phenomena*, John Wiley & Sons, New York, NY, 1960.

Brand, C.W., J.E. Heinemann and K. Aziz. The Grid Orientation Effect in Reservoir Simulation, Paper SPE-21228, presented at Society of Petroleum Engineers Eleventh Symposium on Reservoir Simulation, Anaheim, CA, February 1991.

Cass, A., G.S. Campbell and T.L. Jones. Enhancement of Thermal Water Vapor Diffusion in Soil. *Soil Sci. Soc. Am. J.,* Vol. 48, no. 1, pp. 25 - 32, 1984.

Clarke, M., and Bishnoi, P.R., Determination of activation energy and intrinsic rate constant of methane gas hydrate decomposition; Canadian Journal of Chemical Engineering, v. 79, no.1, p. 143-147, 2001.

Coats, K.H. Geothermal Reservoir Modeling, paper SPE-6892, presented at the 52nd Annual Fall Technical Conference and Exhibition of the SPE, Denver, Colorado, October 1977.

Coats, K.H. and A.B. Ramesh. Effects of Grid Type and Difference Scheme on Pattern Steamflood Simulation Results, paper SPE-11079, presented at the 57th Annual Fall Technical Conference and Exhibition of the Society of Petroleum Engineers, New Orleans, LA, September 1982.

Corey, A.T. The Interrelation Between Gas and Oil Relative Permeabilities, *Producers Monthly*, 38-41, November 1954.

Cygan, R.T. The Solubility of Gases in NaCl Brine and a Critical Evaluation of Available Data, SAND90-2848, Sandia National Laboratories, Albuquerque, New Mexico, 1991.

Dean, J.A. (ed.). *Lange's Handbook of Chemistry*, McGraw Hill, Thirteenth Edition, New York, 1985.

de Marsily, G. *Quantitative Hydrogeology*, Academic Press, Orlando, FL, 1986.

Doughty, C. and K. Pruess. A Similarity Solution for Two-Phase Water, Air and Heat Flow Near a Linear Heat Source in a Porous Medium, *J. of Geophys. Res.*, 97 (B2), 1821-1838, 1992.

Duff, I.S. MA28—A Set of FORTRAN Subroutines for Sparse Unsymmetric Linear Equations, AERE Harwell Report R 8730, 1977.

Edlefsen, N.E. and A.B.C. Anderson. Thermodynamics of Soil Moisture, *Hilgardia*, 15 (2), 31–298, 1943.

Edwards, A.L. TRUMP: A Computer Program for Transient and Steady State Temperature Distributions in Multidimensional Systems, National Technical Information Service, National Bureau of Standards, Springfield, VA, 1972.

Falta, R.W., K. Pruess, I. Javandel and P.A. Witherspoon. Density - Driven Flow of Gas in the Unsaturated Zone Due to the Evaporation of Volatile Organic Compounds, *Water. Resour. Res.*, Vol. 25, No. 10, pp. 2159 - 2169, 1989.

Falta, R.W., K. Pruess, S. Finsterle, and A. Battistelli. T2VOC User's Guide, Lawrence Berkeley Laboratory Report LBL-36400, Berkeley, CA, March 1995.

Fatt, I. and W.A. Klikoff. Effect of Fractional Wettability on Multiphase Flow Through Porous Media, *AIME Transactions*, 216, 246, 1959.

Faust, C.R. and J.W. Mercer. Summary of Our Research in Geothermal Reservoir Simulation, Proc. Workshop on Geothermal Reservoir Engineering, Stanford University, Stanford, CA, SGP-TR-12, 1975.

Finley, N.C. and M. Reeves. Swift Self-Teaching Curriculum, Report SAND81-0410, Sandia National Laboratory, Albuquerque, NM, 1982.

Grant, M.A. Permeability Reduction Factors at Wairakei, paper 77-HT-52, presented at AICHE-ASME Heat Transfer Conference, Salt Lake City, Utah, August 1977.

Hadgu, T. and D.H. Freeston. A Multi-Purpose Wellbore Simulator. *Geothermal Resources Council Transactions*, Vol. 14, pp. 1279 - 1286, 1990.

Hadgu, T., R.W. Zimmerman and G.S. Bodvarsson. Coupled Reservoir-Wellbore Simulation of Geothermal Reservoir Behavior. *Geothermics*, Vol. 24, No. 2, pp. 145-166, 1995.

Herbert, A.W., C.P. Jackson and D.A. Lever. Coupled Groundwater Flow and Solute Transport with Fluid Density Strongly Dependent on Concentration, *Water Resour.*

Res., Vol. 24, No. 10, pp. 1781 - 1795, 1988.

Hirschfelder, J.O., C.F. Curtiss and R.B. Bird. *Molecular Theory of Gases and Liquids*, John Wiley & Sons, New York, NY, 1954.

International Formulation Committee. *A Formulation of the Thermodynamic Properties of Ordinary Water Substance,* IFC Secretariat, Düsseldorf, Germany, 1967.

Itasca Consulting Group. *FLAC3D: Fast Lagrangian Analysis of Continua in 3 Dimensions*, Minneapolis, Minnesota, 2002.

Jury, W.A., W.F. Spencer and W.J. Farmer. Behavior Assessment Model for Trace Organics in Soil: I. Model Description, *J. Environ. Qual.,* Vol. 12, No. 4, pp. 558 - 564, 1983.

Kim, H.C., Bishnoi, P.R., Heideman, R.A., and Rizvi, S.S.H, *Chem. Eng. Sci*., *42*, 1645-1653, 1987.

Klinkenberg, L.J. The Permeability of Porous Media to Liquids and Gases, in *API Drilling and Production Practice,* pp. 200–213, 1941.

Kruger, P. and H.J. Ramey, Jr. Stimulation and Reservoir Engineering of Geothermal Resources, Stanford Geothermal Program Report SGR-TR-1, Stanford University, Stanford, CA, 1974.

Lam, S.T., A. Hunsbedt, P. Kruger and K. Pruess. Analysis of the Stanford Geothermal Reservoir Model Experiments Using the LBL Reservoir Simulator, *Geothermics*, 17 (4), 595 - 605, LBL-25957, 1988.

Leverett, M.C. Capillary Behavior in Porous Solids, *Trans. Soc. Pet. Eng. AIME*, 142, 152-169, 1941.

Loomis, A.G. Solubilities of Gases in Water, in: *International Critical Tables, Vol. III*, E. W. Washburn, (ed.), McGraw-Hill, New York, NY, pp. 255–257, 1928.

Marion, G.M., Jakubowski, S.D., The compressibility of ice to 2.0 kbar, Cold Regions Science and Technology, Volume 38, Issues 2-3, p. 211-218, 2004.

de Marsily, G. *Quantitative Hydrogeology*, Academic Press, Orlando, FL, 1986.

Mason, E.A. and A.P. Malinauskas. *Gas Transport in Porous Media: The Dusty Gas Model,* Elsevier, Amsterdam, The Netherlands, 1983.

Michaelides, E.E. Thermodynamic Properties of Geothermal Fluids, Geothermal Resources Council Transactions, Vol. 5, pp. 361 - 364, 1981.

Miller, A.B. A Brine-Steam Properties Computer Program for Geothermal Energy Calculations, Lawrence Livermore National Laboratory Report UCRL-52495, Livermore, CA, June 1978.

Millington, R.J. and J.P. Quirk. Permeability of Porous Solids, *Trans. Faraday Soc.*, 57, 1200-1207, 1961.

Milly, P.C.D. Moisture and Heat Transport in Hysteretic, Inhomogeneous Porous Media: A Matric-Head Based Formulation and a Numerical Model, *Water Resour. Res.*, Vol. 18, No. 3, pp. 489 - 498, 1982.

Moridis, G.J. Numerical studies of gas production from methane hydrates; Society of Petroleum Engineers Journal, Vol. 32, No.8, pp. 359-370, 2003.

Moridis, G.J., Numerical studies of gas production from methane hydrates; Society of Petroleum Engineers Reservoir Evaluation and Engineering, Vol. 27, No.3, pp. 175-183, 2004.

Moridis, G.J. and T. Collett, Gas Production from Class 1 Hydrate Accumulations, in Recent Advances in the Study of Gas Hydrates, C. Taylor and J. Qwan, Editors, Kluwer Academic/Plenum Publishers (Section I, Chapter 6, pp. 75-88), 2004.

Moridis, G. and K. Pruess. TOUGH Simulations of Updegraff's Set of Fluid and Heat Flow Problems, Lawrence Berkeley Laboratory Report LBL-32611, Berkeley, CA, November 1992.

Moridis, G. and K. Pruess. Flow and Transport Simulations Using T2CG1, a Package of Conjugate Gradient Solvers for the TOUGH2 Family of Codes, Lawrence Berkeley Laboratory Report LBL-36235, Berkeley, CA, 1995.

Moridis, G. and K. Pruess. T2SOLV: An Enhanced Package of Solvers for the TOUGH2 Family of Reservoir Simulation Codes, *Geothermics*, Vol. 27, No. 4, pp. 415 - 444, 1998.

Moridis, G.J., Collett, T. S., Dallimore, S.R., Satoh, T., Hancock, S., and Weatherhill, B. Numerical studies of gas production from several methane hydrate zones at the Mallik site, Mackenzie Delta, Canada; Journal of Petroleum Science and Engineering, v. 43, p. 219-239, 2004 (also Report LBNL-50257, Lawrence Berkeley National Laboratory, Berkeley, California, 2002).

Moridis, G.J., Kowalsky, M.B., Pruess, K., TOUGH-Fx/HYDRATE v1.1 User's Manual: A Code for the Simulation of System Behavior in Hydrate-Bearing Geologic Media, Report LBNL-58950, Lawrence Berkeley National Laboratory, Berkeley, CA (2005).

Moridis, G.J., Kowalsky, M.B., Pruess, K., TOUGH+HYDRATE v1.0 User's Manual: A Code for the Simulation of System Behavior in Hydrate-Bearing Geologic Media, Report LBNL-0149E, Lawrence Berkeley National Laboratory, Berkeley, CA (2008).

Morrow, C., D. Lockner, D. Moore and J. Byerlee. Permeability of Granite in a Temperature Gradient, *Journal of Geophysical Research*, 86 (84), 3002-3008, April 1981.

Mualem, Y. A New Model for Predicting the Hydraulic Conductivity of Unsaturated Porous Media, *Water Resour. Res.*, Vol. 12, No. 3, pp. 513 - 522, 1976.

Murray, L. and C. Gunn. Toward Integrating Geothermal Reservoir and Wellbore Simulation: TETRAD and WELLSIM, Proc. 15th NZ Geothermal Workshop, Auckland, New Zealand, 1993.

Narasimhan, T.N. and P.A. Witherspoon. An Integrated Finite Difference Method for Analyzing Fluid Flow in Porous Media, *Water Resour. Res.*, Vol. 12, No. 1, pp. 57 – 64, 1976.

Narasimhan, T.N., P.A. Witherspoon and A.L. Edwards. Numerical Model for Saturated-Unsaturated Flow in Deformable Porous Media, Part 2: The Algorithm, *Water Resour. Res.*, 14 (2), 255-261, 1978.

Oldenburg, C.M. and K. Pruess. A Two-Dimensional Dispersion Module for the TOUGH2 Simulator, Lawrence Berkeley Laboratory Report LBL-32505, Berkeley, CA, September 1993.

Oldenburg, C.M. and K. Pruess. EOS7R: Radionuclide Transport for TOUGH2, Lawrence Berkeley Laboratory Report LBL-34868, Berkeley, CA, November 1995.

Oldenburg, C.M. and K. Pruess. Higher-Order Differencing for Phase Front Propagation in Geothermal Systems, Proc., Twenty-Third Workshop on Geothermal Reservoir Engineering, Stanford University, January 1998.

O'Sullivan, M.J. A Similarity Method for Geothermal Well Test Analysis, *Water Resour. Res.*, Vol. 17, No. 2, pp. 390 – 398, 1981.

O'Sullivan, M.J., G.S. Bodvarsson, K. Pruess and M.R. Blakeley. Fluid and Heat Flow in Gas-Rich Geothermal Reservoirs, *Society of Petroleum Engineers Journal*, Vol. 25, No. 2, pp. 215 - 226, April 1985.

Pape, H., C. Clauser and J. Iffland. Permeability Prediction Based on Fractal Pore-Space Geometry, *Geophysics,* Vol. 64, No. 5, pp. 1447 - 1460, 1999.

Peaceman, D.W. *Fundamentals of Numerical Reservoir Simulation*, Elsevier, Amsterdam, The Netherlands, 1977.

Peaceman, D.W. Interpretation of Well-Block Pressures in Numerical Reservoir Simulation with Nonsquare Grid Blocks and Anisotropic Permeability, paper SPE-10528, presented at the Sixth SPE Symposium on Reservoir Simulation of the Society of Petroleum Engineers, New Orleans, LA, January 1982.

Phillips, O.M. *Flow and Reactions in Permeable Rocks,* Cambridge University Press, Cambridge, New York, Melbourne, 1991.

Pickens, J.F., R.W. Gillham and D.R. Cameron. Finite Element Analysis of the Transport of Water and Solutes in Tile-Drained Soils, *J. of Hydrology*, 40, 243-264, 1979.

Potter, J.M. and D.L. Brown. The Volumetric Properties of Aqueous Sodium Chloride Solutions, U. S. Geological Survey, Bulletin 1421-C, 1977.

Pritchett, J.W. Numerical Calculation of Multiphase Fluid and Heat Flow in Hydrothermal Reservoirs, Proc. Workshop on Geothermal Reservoir Engineering, Stanford University, Stanford, CA, pp. 201–205, SGP-TR-12, 1975.

Pritchett, J.W., M.H. Rice and T.D. Riney. Equation-of-State for Water-Carbon Dioxide Mixtures: Implications for Baca Reservoir, Report DOE/ET/27163-8, Systems, Science and Software, La Jolla, CA, February 1981.

Pruess, K. GMINC - A Mesh Generator for Flow Simulations in Fractured Reservoirs, Lawrence Berkeley Laboratory Report LBL-15227, Berkeley, CA, March 1983.

Pruess, K. TOUGH User's Guide, Nuclear Regulatory Commission Report NUREG/CR-4645; also Lawrence Berkeley Laboratory Report LBL-20700, 1987.

Pruess, K. TOUGH2 - A General Purpose Numerical Simulator for Multiphase Fluid and Heat Flow, Lawrence Berkeley Laboratory Report LBL-29400, Berkeley, CA, 1991a.

Pruess, K. EOS7, An Equation-of-State Module for the TOUGH2 Simulator for Two-Phase Flow of Saline Water and Air, Lawrence Berkeley Laboratory Report LBL-31114, Berkeley, CA, 1991b.

Pruess, K. Grid Orientation and Capillary Pressure Effects in the Simulation of Water Injection into Depleted Vapor Zones, *Geothermics*, Vol. 20, No. 5/6, pp. 257 - 277, 1991c.

Pruess, K., J.M. Zerzan, R.C. Schroeder and P.A. Witherspoon. Description of the Three-Dimensional Two-Phase Simulator SHAFT78 for Use in Geothermal Reservoir Studies, paper SPE-7699, presented at the Fifth SPE Symposium on Reservoir Simulation, Denver, CO, February 1979.

Pruess, K. and R.C. Schroeder. SHAFT 79 User's Manual, Lawrence Berkeley Laboratory Report LBL-10861, Berkeley, CA, March 1980.

Pruess, K., and T.N. Narasimhan. On Fluid Reserves and the Production of Superheated Steam from Fractured, Vapor-Dominated Geothermal Reservoirs, *J. Geophys. Res.*,

Vol. 87, No. B11, pp. 9329 - 9339, 1982.

Pruess, K. and G.S. Bodvarsson. A Seven-Point Finite Difference Method for Improved Grid Orientation Performance in Pattern Steam Floods, *Proceedings*, Seventh Society of Petroleum Engineers Symposium on Reservoir Simulation, Paper SPE-12252, pp. 175 - 184, San Francisco, CA, 1983.

Pruess, K. and G.S. Bodvarsson. Thermal Effects of Reinjection in Geothermal Reservoirs with Major Vertical Fractures, *J. Pet. Tech.*, Vol. 36, No. 10, pp. 1567 – 1578, 1984.

Pruess, K. and T.N. Narasimhan. A Practical Method for Modeling Fluid and Heat Flow in Fractured Porous Media, *Soc. Pet. Eng. J.*, Vol. 25, No. 1, pp. 14-26, February 1985.

Pruess, K., J.S.Y. Wang and Y.W. Tsang. On Thermohydrological Conditions Near High-Level Nuclear Wastes Emplaced in Partially Saturated Fractured Tuff. Part 2. Effective Continuum Approximation, *Water Resour. Res.*, Vol. 26, No. 6, pp. 1249 - 1261, 1990.

Pruess, K., A. Simmons, Y.S. Wu and G. Moridis. TOUGH2 Software Qualification, Lawrence Berkeley National Laboratory Report LBL-38383, Berkeley, CA, February 1996.

Pruess, K., C. Oldenburg, and G.Moridis. TOUGH2 User's Guide – Version 2.0, Lawrence Berkeley Laboratory Report LBL-43134, Berkeley, CA, 1999.

Reeves, M., D. S. Ward, N. D. Johns and R. M. Cranwell. Theory and Implementation of Swift II, The Sandia Waste-Isolation Flow and Transport Model for Fractured Media, Report No. SAND83-1159, Sandia National Laboratory, Albuquerque, NM, 1986.

Richards, L.A. Capillary Conduction of Liquids Through Porous Mediums, *Physics*, Vol. 1, pp. 318 - 333, 1931.

Rutqvist J. and C.-F Tsang. A Study of Caprock Hydromechanical Changes Associated with CO_2 Injection into a Brine Aquifer. *Environmental Geology*, Vol. 42, pp. 296-305, 2002.

Scheidegger, A. E. *The Physics of Flow Through Porous Media*, University of Toronto Press, Toronto and Buffalo, Third Edition, 1974.

Sleijpen, G.L.G. and D. Fokkema. BiCGSTAB(m) for Linear Equations Involving Unsymmetric Matrices with Complex Spectrum, *Electronic Transactions on Numerical Analysis*, Vol. 1, pp. 11 - 32, 1993.

Sloan, E.D., *Clathrate Hydrates of Natural Gases*, Marcel Dekker, Inc.: New York, NY, 1998.

Stone, H.L., Probability Model for Estimating Three-Phase Relative Permeability. *Trans. SPE of AIME,* Vol. 249, 214-218, 1970.

Sutton, F.M. and A. McNabb. Boiling Curves at Broadlands Field, New Zealand, *N. Z. J. Sci.*, Vol. 20, pp. 333 - 337, 1977.

Tanaka, H., Thermal expansivities of cubic ice I and ice VII, Journal of Molecular Structure: THEOCHEM, Volumes 461-462, Pages 561-567, 1999.

TecPlot, Inc., *Tecplot 10 User's Manual*, Bellevue, Washington, 2003.

Thomas, G.W. *Principles of Hydrocarbon Reservoir Simulation*. International Human Resources Development Corporation, Boston, 1982.

Tsang, Y.W. and K. Pruess. Further Modeling Studies of Gas Movement and Moisture Migration at Yucca Mountain, Nevada, Lawrence Berkeley Laboratory Report LBL-29127, Berkeley, CA, 1990.

Udell, K.S. and J.S. Fitch. Heat and Mass Transfer in Capillary Porous Media Considering Evaporation, Condensation, and Non-Condensible Gas Effects, paper presented at 23rd ASME/AIChE National Heat Transfer Conference, Denver, CO, 1985.

van der Vorst, H.A. Bi-CGSTAB: A Fast and Smoothly Converging Variant of BiCG in the Presence of Rounding Errors, *SIAM J. Sci. Statist. Comput.*, Vol. 13, pp. 631 - 644, 1992.

van Genuchten, M.Th. A Closed-Form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils, *Soil Sci. Soc.*, Vol. 44, pp. 892 - 898, 1980.

Vargaftik, N.B. *Tables on the Thermophysical Properties of Liquids and Gases*, 2nd Ed., John Wiley & Sons, New York, NY, 1975.

Vaughan, P.J. Analysis of Permeability Reduction During Flow of Heated, Aqueous Fluid Through Westerly Granite, in C.F. Tsang (ed.), *Coupled Processes Associated with Nuclear Waste Repositories*, pp. 529 - 539, Academic Press, New York, 1987.

Verma, A.K., K. Pruess, C.F. Tsang and P.A. Witherspoon. A Study of Two-Phase Concurrent Flow of Steam and Water in an Unconsolidated Porous Medium, Proc. 23rd National Heat Transfer Conference, Am. Society of Mechanical Engineers, Denver, CO, 135–143, 1985.

Verma, A. and K. Pruess. Thermohydrologic Conditions and Silica Redistribution Near High-Level Nuclear Wastes Emplaced in Saturated Geological Formations, *J. of Geophys. Res.*, Vol. 93, No. B2, pp. 1159-1173, 1988.

Vinsome, P.K.W. and J. Westerveld. A Simple Method for Predicting Cap and Base

Rock Heat Losses in Thermal Reservoir Simulators, *J. Canadian Pet. Tech.*, Vol. 19, No. 3, pp. 87 – 90, July-September 1980.

Walker, W.R., J.D. Sabey, and D.R. Hampton. Studies of Heat Transfer and Water Migration in Soils, Final Report, Department of Agricultural and Chemical Engineering, Colorado State University, Fort Collins, CO, 80523, April 1981.

Warren, J.E. and P.J. Root. The Behavior of Naturally Fractured Reservoirs, *Soc. Pet. Eng. J., Transactions, AIME, 228*, 245-255, September 1963.

Webb, S.W. Gas-Phase Diffusion in Porous Media - Evaluation of an Advective-Dispersive Formulation and the Dusty Gas Model for Binary Mixtures, *J. Por. Media,* Vol. 1, No. 2, pp. 187 - 199, 1998.

Webb, S.W. and C.K. Ho. Review of Enhanced Vapor Diffusion in Porous Media, in: K. Pruess (ed.), *Proceedings of the TOUGH Workshop '98,* Lawrence Berkeley National Laboratory Report LBNL-41995, pp. 257 - 262, Berkeley, CA, 1998a.

Webb, S.W. and C.K. Ho. Enhanced Vapor Diffusion in Porous Media - LDRD Final Report, Sandia National Laboratories Report SAND98-2772, Albuquerque, NM, 1998b.

Wu, Y.S., K. Pruess and Z.X. Chen. Buckley-Leverett Flow in Composite Media, *SPE Advanced Technology Series,* Vol. 1, No. 2, pp. 36 - 42, 1993.

Xu, T., Y. Ontoy, P. Molling, N. Spycher, M. Parini and K. Pruess. Reactive Transport Modeling of Injection Well Scaling and Acidizing at Tiwi Field, Philippines, *Geothermics,* Vol. 33, No. 4, pp. 477 - 491, 2004.

Yanosik, J.L. and T.A. McCracken. A Nine-Point, Finite Difference Reservoir Simulator for Realistic Prediction of Adverse Mobility Ratio Displacements, *Soc. Pet. Eng. J.*, 253 – 262, 1979.

Appendix A.

A1. Input File: `Test_1T`, Output File: `Hydrate_Info_1T`

Time(d)	CH4MassRate(kg/s)	CH4VolRate(m3/s)	Cum_CH4_Mass(kg)	Cum_CH4_Vol(m3)	FreeCH4inRes(m3)	RemainHydr(kg)
1.1574074E-03	1.6198630E-05	2.3929264E-05	1.6198630E-03	2.3929264E-03	0.0000000E+00	1.3802322E+02
3.4722222E-03	1.9806390E-04	2.9258792E-04	4.1232644E-02	6.0910511E-02	5.1621937E-02	1.3771672E+02
5.7870370E-03	8.6736483E-04	1.2813060E-03	2.1470561E-01	3.1717171E-01	1.4044911E-01	1.3637447E+02
8.1018519E-03	8.4141420E-04	1.2429707E-03	3.8298845E-01	5.6576586E-01	1.7939573E-01	1.3507238E+02
1.2731481E-02	8.0006981E-04	1.1818951E-03	7.0301637E-01	1.0385239E+00	2.1142198E-01	1.3259616E+02
1.7361111E-02	8.2692531E-04	1.2215672E-03	1.0337865E+00	1.5271508E+00	2.2629400E-01	1.3003682E+02
1.9675926E-02	8.3668691E-04	1.2359874E-03	1.2011239E+00	1.7743482E+00	2.3104016E-01	1.2874204E+02
2.4305556E-02	7.9581778E-04	1.1756139E-03	1.5194510E+00	2.2445938E+00	2.3472764E-01	1.2627898E+02
3.3564815E-02	1.1309235E-04	1.6706454E-04	1.6099249E+00	2.3782454E+00	1.6343672E-01	1.2557894E+02
4.2824074E-02	1.8671387E-05	2.7582120E-05	1.6248620E+00	2.4003111E+00	1.3725651E-01	1.2546336E+02
5.2083333E-02	8.8955489E-05	1.3140861E-04	1.6960264E+00	2.5054380E+00	1.9879142E-01	1.2491272E+02
6.1342593E-02	9.5475325E-05	1.4103997E-04	1.7724066E+00	2.6182700E+00	2.4322878E-01	1.2432173E+02
7.0601852E-02	1.0209346E-04	1.5081654E-04	1.8540814E+00	2.7389232E+00	2.7202045E-01	1.2368977E+02
8.9120370E-02	1.3021400E-04	1.9235733E-04	2.0624238E+00	3.0466949E+00	3.1396067E-01	1.2207772E+02
1.2615741E-01	2.0194265E-04	2.9831776E-04	2.7086403E+00	4.0013118E+00	4.1820952E-01	1.1707760E+02
1.6319444E-01	1.6646184E-04	2.4590408E-04	3.2413182E+00	4.7882048E+00	4.5955191E-01	1.1295600E+02
2.0023148E-01	2.6045029E-05	3.8474758E-05	3.3246622E+00	4.9113241E+00	3.9170954E-01	1.1231112E+02
2.0949074E-01	3.1260689E-05	4.6179541E-05	3.3496708E+00	4.9482677E+00	3.8938563E-01	1.1211762E+02
2.2800926E-01	4.1963326E-05	6.1989904E-05	3.4168121E+00	5.0474515E+00	4.0823841E-01	1.1159811E+02
2.6504630E-01	8.7667406E-05	1.2950580E-04	3.6973478E+00	5.4618701E+00	5.1997370E-01	1.0942746E+02
3.0208333E-01	1.0187603E-04	1.5049535E-04	4.0233511E+00	5.9434552E+00	5.9897146E-01	1.0690501E+02
3.7615741E-01	1.1184332E-04	1.6521942E-04	4.7391483E+00	7.0008595E+00	6.9038263E-01	1.0136651E+02
3.9467593E-01	1.0694761E-04	1.5798729E-04	4.9102645E+00	7.2536392E+00	7.0453189E-01	1.0004250E+02
4.1319444E-01	2.2478279E-06	3.3205813E-06	4.9138610E+00	7.2589521E+00	6.0121099E-01	1.0001467E+02
4.3171296E-01	1.1315731E-05	1.6716051E-05	4.9319662E+00	7.2856978E+00	5.5779256E-01	9.9874581E+01
4.6875000E-01	2.7842171E-05	4.1129569E-05	5.0210612E+00	7.4173124E+00	5.7269639E-01	9.9185207E+01
5.0578704E-01	5.1548642E-05	7.6149716E-05	5.1860168E+00	7.6609915E+00	6.5666086E-01	9.7908860E+01
5.7986111E-01	6.4386266E-05	9.5113968E-05	5.5980889E+00	8.2697209E+00	7.8133740E-01	9.4720444E+01
6.5393519E-01	7.1149393E-05	1.0510473E-04	6.0534450E+00	8.9423911E+00	8.6434726E-01	9.1197118E+01
7.2800926E-01	7.3882782E-05	1.0914260E-04	6.5262948E+00	9.6409038E+00	9.2417100E-01	8.7538434E+01
7.3726852E-01	7.2005861E-05	1.0636994E-04	6.5838995E+00	9.7259997E+00	9.2969719E-01	8.7092717E+01
7.5578704E-01	3.0696349E-06	4.5345875E-06	6.5888109E+00	9.7332551E+00	8.4143783E-01	8.7054715E+01
7.6041667E-01	5.6985965E-07	8.4181946E-07	6.5890389E+00	9.7335918E+00	8.2054186E-01	8.7052951E+01
7.6967593E-01	8.8582290E-06	1.3085730E-05	6.5961255E+00	9.7440604E+00	7.9476498E-01	8.6998119E+01
7.8819444E-01	1.8187453E-05	2.6867233E-05	6.6252254E+00	9.7870480E+00	7.7762718E-01	8.6772957E+01
8.2523148E-01	3.0282818E-05	4.4734990E-05	6.7221304E+00	9.9301999E+00	8.0740000E-01	8.6023153E+01
8.9930556E-01	4.1604366E-05	6.1459634E-05	6.9883984E+00	1.0323542E+01	9.1044921E-01	8.3962900E+01
9.7337963E-01	4.7372857E-05	6.9981079E-05	7.2915846E+00	1.0771420E+01	9.9580328E-01	8.1616991E+01
1.1215278E+00	5.7719382E-05	8.5265379E-05	8.0303927E+00	1.1862817E+01	1.1427281E+00	7.5900450E+01
1.2696759E+00	3.8068717E-05	5.6236632E-05	8.5176723E+00	1.2582646E+01	1.1722888E+00	7.2130116E+01
1.4178241E+00	3.4832462E-05	5.1455906E-05	8.9635278E+00	1.3241282E+01	1.2392836E+00	6.8680301E+01
1.4548611E+00	3.9906662E-05	5.8951717E-05	9.0912291E+00	1.3429927E+01	1.2737760E+00	6.7692209E+01
1.5289352E+00	4.3869598E-05	6.4805926E-05	9.3719946E+00	1.3844685E+01	1.3470740E+00	6.5519782E+01
1.5381944E+00	4.4169907E-05	6.5249553E-05	9.4073305E+00	1.3896885E+01	1.3560778E+00	6.5246369E+01
1.5428241E+00	4.4302632E-05	6.5445621E-05	9.4250515E+00	1.3923063E+01	1.3605329E+00	6.5109252E+01
1.5520833E+00	4.6065006E-05	6.8049070E-05	9.4619035E+00	1.3977502E+01	1.3707260E+00	6.4824109E+01
1.5613426E+00	4.7680689E-05	7.0435822E-05	9.5000481E+00	1.4033851E+01	1.3818638E+00	6.4528965E+01

Time(d)	CH4MassRate(kg/s)	CH4VolRate(m3/s)	Cum_CH4_Mass(kg)	Cum_CH4_Vol(m3)	FreeCH4inRes(m3)	RemainHydr(kg)
1.5798611E+00	4.8980821E-05	7.2356429E-05	9.5784174E+00	1.4149621E+01	1.4038943E+00	6.3922581E+01
1.6168981E+00	5.0581524E-05	7.4721050E-05	9.7402783E+00	1.4388729E+01	1.4454965E+00	6.2670180E+01
1.6909722E+00	5.1918733E-05	7.6696429E-05	1.0072558E+01	1.4879586E+01	1.5182220E+00	6.0099158E+01

1.8391204E+00	3.3319107E-05	4.9220318E-05	1.0499043E+01	1.5509606E+01	1.5417515E+00	5.6799226E+01
1.9872685E+00	3.2659611E-05	4.8246084E-05	1.0917086E+01	1.6127156E+01	1.6139922E+00	5.3564611E+01
2.1354167E+00	3.9428248E-05	5.8244985E-05	1.1421767E+01	1.6872692E+01	1.7425356E+00	4.9659628E+01
2.4317130E+00	3.9314755E-05	5.8077329E-05	1.2428225E+01	1.8359471E+01	1.9126949E+00	4.1872143E+01
2.5462963E+00	2.1553721E-05	3.1840019E-05	1.2641607E+01	1.8674687E+01	1.8545042E+00	4.0221098E+01

A2. Input File: `Test_1T`, Output File: `Conx_Time_Series_1T`

Time [days]	GasFlow [kg/s]	AquFlow [kg/s]	HeatFlow [W]	CH4inGasFlow [m^3/s]	TotalCH4Flow [m^3/s]	CumCH4Vol [m^3]	CumH2OMass [kg]	X_inhibitor [kg/kg]
1.15741E-03	0.0000000E+00	2.6712412E-06	-3.5073896E+03	0.0000000E+00	4.4602087E-09	4.4602087E-07	2.6682219E-04	0.0000000E+00
3.47222E-03	8.3866903E-06	5.0874594E-03	-3.1950513E+03	1.2386278E-05	2.2141995E-05	4.4288451E-03	1.0164379E+00	0.0000000E+00
5.78704E-03	5.5737655E-04	7.9253865E-03	-3.0595598E+03	8.2318800E-04	8.3842878E-04	1.7211460E-01	2.5994518E+00	0.0000000E+00
8.10185E-03	7.0040564E-04	3.0635278E-03	-3.1218690E+03	1.0344270E-03	1.0403010E-03	3.8017480E-01	3.2113621E+00	0.0000000E+00
1.27315E-02	7.3671736E-04	4.9582931E-04	-3.1055453E+03	1.0880555E-03	1.0890049E-03	8.1577677E-01	3.4094367E+00	0.0000000E+00
1.73611E-02	7.9218267E-04	5.4820931E-04	-3.1197591E+03	1.1699721E-03	1.1699721E-03	1.2837656E+00	3.6287205E+00	0.0000000E+00
1.96759E-02	8.1083510E-04	8.6930306E-04	-3.1038478E+03	1.1975198E-03	1.1975198E-03	1.5232695E+00	3.8025811E+00	0.0000000E+00
2.43056E-02	7.8028440E-04	1.1196924E-03	-2.8792784E+03	1.1523995E-03	1.1523995E-03	1.9842294E+00	4.2504580E+00	0.0000000E+00
3.35648E-02	1.7391226E-04	2.0039365E-03	-2.4704509E+03	2.5682605E-04	2.5682605E-04	2.1896902E+00	5.8536072E+00	0.0000000E+00
4.28241E-02	4.2511330E-05	6.9881458E-04	-1.9892105E+03	6.2771977E-05	6.2771977E-05	2.2399078E+00	6.4126589E+00	0.0000000E+00
5.20833E-02	3.8307515E-05	1.5436968E-03	-1.5601622E+03	5.6559161E-05	5.8757888E-05	2.2869141E+00	7.6464256E+00	0.0000000E+00
6.13426E-02	5.8057453E-05	1.1343065E-03	-1.4202163E+03	8.5710979E-05	8.7244924E-05	2.3567100E+00	8.5530401E+00	0.0000000E+00
7.06019E-02	7.7076020E-05	6.8600408E-04	-1.3239515E+03	1.1377850E-04	1.1466963E-04	2.4484457E+00	9.1013608E+00	0.0000000E+00
8.91204E-02	1.1108496E-04	4.0454306E-04	-1.1731070E+03	1.6396014E-04	1.6445936E-04	2.7115807E+00	9.7480890E+00	0.0000000E+00
1.26157E-01	1.7837436E-04	4.0007534E-04	-9.9299633E+02	2.6323725E-04	2.6370758E-04	3.5554450E+00	1.1027311E+01	0.0000000E+00
1.63194E-01	1.5662375E-04	2.4469666E-05	-9.0746418E+02	2.3110641E-04	2.3113417E-04	4.2950743E+00	1.1105554E+01	0.0000000E+00
2.00231E-01	4.0592867E-05	4.9356761E-04	-8.9300083E+02	5.9887809E-05	5.9887809E-05	4.4867153E+00	1.2684970E+01	0.0000000E+00
2.09491E-01	3.3370656E-05	1.0044957E-04	-7.9308809E+02	4.9231061E-05	4.9231061E-05	4.5261001E+00	1.2765330E+01	0.0000000E+00
2.28009E-01	3.3822471E-05	2.1696273E-04	-7.0225541E+02	4.9894962E-05	5.0129255E-05	4.6063069E+00	1.3112217E+01	0.0000000E+00
2.65046E-01	6.3748439E-05	6.5123144E-04	-5.8122648E+02	9.4035116E-05	9.4729692E-05	4.9094419E+00	1.5194653E+01	0.0000000E+00
3.02083E-01	8.4577323E-05	3.9556326E-04	-5.6835957E+02	1.2475177E-04	1.2516924E-04	5.3099835E+00	1.6459551E+01	0.0000000E+00
3.76157E-01	1.0131929E-04	1.3977721E-04	-5.4261451E+02	1.4943287E-04	1.4957837E-04	6.2672851E+00	1.7353495E+01	0.0000000E+00
3.94676E-01	1.0013071E-04	2.6047610E-05	-5.4491268E+02	1.4767653E-04	1.4770356E-04	6.5036108E+00	1.7395141E+01	0.0000000E+00
4.13194E-01	4.5661452E-05	1.4241949E-03	-7.9923094E+02	6.7340173E-05	6.7340173E-05	6.6113550E+00	1.9673853E+01	0.0000000E+00
4.31713E-01	2.9657736E-05	6.1874451E-04	-6.2945266E+02	4.3737132E-05	4.3737132E-05	6.6813345E+00	2.0663844E+01	0.0000000E+00
4.68750E-01	2.4761561E-05	6.7312991E-05	-4.8154999E+02	3.6515463E-05	3.6584268E-05	6.7984041E+00	2.0879097E+01	0.0000000E+00
5.05787E-01	3.3784973E-05	5.1327624E-04	-3.9848024E+02	4.9821246E-05	5.0344679E-05	6.9595071E+00	2.2520447E+01	0.0000000E+00
5.79861E-01	5.0874127E-05	3.4132573E-04	-3.9204563E+02	7.5018177E-05	7.5364058E-05	7.4418370E+00	2.4703433E+01	0.0000000E+00
6.53935E-01	6.1848595E-05	1.8049967E-04	-3.8816008E+02	9.1196697E-05	9.1378549E-05	8.0266598E+00	2.5857843E+01	0.0000000E+00
7.28009E-01	6.6972468E-05	9.1628461E-05	-3.7954474E+02	9.8748164E-05	9.8840056E-05	8.6592361E+00	2.6443867E+01	0.0000000E+00
7.37269E-01	6.6765836E-05	3.9951736E-05	-3.8504512E+02	9.8442993E-05	9.8483036E-05	8.7380225E+00	2.6475807E+01	0.0000000E+00
7.55787E-01	3.9990048E-05	1.2253599E-03	-6.1463917E+02	5.8961835E-05	5.8961835E-05	8.8323615E+00	2.8436383E+01	0.0000000E+00
7.60417E-01	3.5566758E-05	1.1651797E-03	-6.0023134E+02	5.2439761E-05	5.2439761E-05	8.8533374E+00	2.8902455E+01	0.0000000E+00
7.69676E-01	3.0562918E-05	7.3786628E-04	-5.1425728E+02	4.5061705E-05	4.5061705E-05	8.8893868E+00	2.9492748E+01	0.0000000E+00
7.88194E-01	2.5563476E-05	2.7101802E-04	-4.2022944E+02	3.7690220E-05	3.7690220E-05	8.9496911E+00	2.9926377E+01	0.0000000E+00
8.25231E-01	2.4151637E-05	1.6174190E-04	-3.3822224E+02	3.5608417E-05	3.5769442E-05	9.0641533E+00	3.0443602E+01	0.0000000E+00
8.99306E-01	3.0629389E-05	3.0179631E-04	-3.0505650E+02	4.5158312E-05	4.5458255E-05	9.3550861E+00	3.2373799E+01	0.0000000E+00
9.73380E-01	3.8084280E-05	2.3084400E-04	-3.0252346E+02	5.6148087E-05	5.6376922E-05	9.7158985E+00	3.3850209E+01	0.0000000E+00
1.12153E+00	4.9593494E-05	1.6632449E-04	-2.8954745E+02	7.3113316E-05	7.3277479E-05	1.0653850E+01	3.5977740E+01	0.0000000E+00
1.26968E+00	3.6418563E-05	7.6955320E-06	-2.9774750E+02	5.3687873E-05	5.3687873E-05	1.1341055E+01	3.6076243E+01	0.0000000E+00
1.41782E+00	3.1277011E-05	6.3511385E-05	-2.7057109E+02	4.6106912E-05	4.6169125E-05	1.1932020E+01	3.6888649E+01	0.0000000E+00
1.45486E+00	3.2553125E-05	1.8020144E-04	-2.4817032E+02	4.7987893E-05	4.8164335E-05	1.2086146E+01	3.7464912E+01	0.0000000E+00
1.52894E+00	3.5970959E-05	1.8943323E-04	-2.4064178E+02	5.3025714E-05	5.3211004E-05	1.2426696E+01	3.8676482E+01	0.0000000E+00
1.53819E+00	3.6391285E-05	1.8471248E-04	-2.4067277E+02	5.3645254E-05	5.3825903E-05	1.2469757E+01	3.8824154E+01	0.0000000E+00
1.54282E+00	3.6596922E-05	1.8184878E-04	-2.4076538E+02	5.3948351E-05	5.4126186E-05	1.2491407E+01	3.8896845E+01	0.0000000E+00
1.55208E+00	3.7282704E-05	2.1344152E-04	-2.3463364E+02	5.4959215E-05	5.5167922E-05	1.2535542E+01	3.9067485E+01	0.0000000E+00
1.56134E+00	3.8108695E-05	2.3751637E-04	-2.2986548E+02	5.6176770E-05	5.6408995E-05	1.2580669E+01	3.9257373E+01	0.0000000E+00
1.57986E+00	3.9508451E-05	2.3187521E-04	-2.2947705E+02	5.8240047E-05	5.8466707E-05	1.2674216E+01	3.9628128E+01	0.0000000E+00
1.61690E+00	4.1614573E-05	2.1037691E-04	-2.3014264E+02	6.1344394E-05	6.1549932E-05	1.2871175E+01	4.0300889E+01	0.0000000E+00
1.69097E+00	4.4035728E-05	1.6965770E-04	-2.3090716E+02	6.4912704E-05	6.5078272E-05	1.3287676E+01	4.1385980E+01	0.0000000E+00
1.83912E+00	3.2135622E-05	1.2779135E-05	-2.4901988E+02	4.7369484E-05	4.7369484E-05	1.3894006E+01	4.1549553E+01	0.0000000E+00
1.98727E+00	2.8900132E-05	7.4857909E-05	-2.2510513E+02	4.2599485E-05	4.2672202E-05	1.4440210E+01	4.2507105E+01	0.0000000E+00
2.13542E+00	3.2575584E-05	1.6331660E-04	-2.0290587E+02	4.8016603E-05	4.8175067E-05	1.5056851E+01	4.4596184E+01	0.0000000E+00
2.43171E+00	3.4722182E-05	8.7176776E-05	-2.0051157E+02	5.1179150E-05	5.1263493E-05	1.6369196E+01	4.6826448E+01	0.0000000E+00
2.54630E+00	2.5556900E-05	1.6205243E-04	-2.3784955E+02	3.7669117E-05	3.7669117E-05	1.6742120E+01	4.8430767E+01	0.0000000E+00

A3. Input File: Test_1P_Ice, Output File: Hydrate_Info_1P_Ice

Time(d)	CH4MassRate(kg/s)	CH4VolRate(m3/s)	Cum_CH4_Mass(kg)	Cum_CH4_Vol(m3)	FreeCH4inRes(m3)	RemainHydr(kg)
1.1574074E-05	5.0165610E-02	7.4106646E-02	5.0165610E-02	7.4106646E-02	7.0446214E-02	1.3770729E+02
2.3148148E-05	6.0252231E-02	8.9007006E-02	1.1041784E-01	1.6311365E-01	1.2021533E-01	1.3724108E+02
4.6296296E-05	9.3022777E-02	1.3741697E-01	2.9646340E-01	4.3794759E-01	1.7736137E-01	1.3580155E+02
6.9444444E-05	1.0042180E-01	1.4834709E-01	4.9730699E-01	7.3464177E-01	2.1233770E-01	1.3424752E+02
1.1574074E-04	1.3830159E-01	2.0430463E-01	1.0505133E+00	1.5518603E+00	2.8619734E-01	1.2996708E+02
1.6203704E-04	1.9347657E-01	2.8581133E-01	1.8244196E+00	2.6951056E+00	3.4578935E-01	1.2397896E+02
2.5462963E-04	4.9666906E-02	7.3369941E-02	2.2217549E+00	3.2820652E+00	3.6210129E-01	1.2090458E+02
3.4722222E-04	2.9435584E-02	4.3483423E-02	2.4572395E+00	3.6299325E+00	4.1364186E-01	1.1908251E+02
4.3981481E-04	2.6480570E-02	3.9118157E-02	2.6690841E+00	3.9428778E+00	4.7149585E-01	1.1744336E+02
6.2500000E-04	3.1574790E-02	4.6643543E-02	3.1742807E+00	4.6891745E+00	5.9263214E-01	1.1353439E+02
8.1018519E-04	4.5579739E-02	6.7332214E-02	3.9035566E+00	5.7664899E+00	7.0906975E-01	1.0789160E+02
1.1805556E-03	2.9693884E-02	4.3864993E-02	4.8537609E+00	7.1701697E+00	8.5491872E-01	1.0053938E+02
1.5509259E-03	2.4406276E-02	3.6053927E-02	5.6347617E+00	8.3238954E+00	9.8674448E-01	9.4496374E+01
1.6435185E-03	2.6715243E-02	3.9464825E-02	5.8484836E+00	8.6396140E+00	1.0278167E+00	9.2842697E+01
1.8287037E-03	3.1566288E-02	4.6630984E-02	6.3535442E+00	9.3857097E+00	1.1039051E+00	8.8934781E+01
2.1990741E-03	2.0837084E-02	3.0781374E-02	7.0203309E+00	1.0370714E+01	1.1598860E+00	8.3775507E+01
2.5694444E-03	1.6042243E-02	2.3698244E-02	7.5336827E+00	1.1129057E+01	1.2163134E+00	7.9803438E+01
2.9398148E-03	1.9843311E-02	2.9313333E-02	8.1686687E+00	1.2067084E+01	1.3086283E+00	7.4890223E+01
3.6805556E-03	1.8948907E-02	2.7992083E-02	9.3813987E+00	1.3858577E+01	1.3874231E+00	6.5506703E+01
4.4212963E-03	2.1836349E-02	3.2257528E-02	1.0778925E+01	1.5923059E+01	1.4520513E+00	5.4693318E+01
5.1620370E-03	1.7714242E-02	2.6168187E-02	1.1912637E+01	1.7597823E+01	1.4055068E+00	4.5921205E+01
5.9027778E-03	1.8265193E-02	2.6982073E-02	1.3081609E+01	1.9324676E+01	1.4357104E+00	3.6876260E+01
6.6435185E-03	1.8558665E-02	2.7415603E-02	1.4269363E+01	2.1079275E+01	1.5082680E+00	2.7685988E+01
7.3842593E-03	1.8405458E-02	2.7189279E-02	1.5447313E+01	2.2819388E+01	1.5970675E+00	1.8571584E+01
8.1250000E-03	1.7411668E-02	2.5721212E-02	1.6561660E+01	2.4465546E+01	1.6399301E+00	9.9493059E+00
8.8657407E-03	1.5006528E-02	2.2168244E-02	1.7522077E+01	2.5884314E+01	1.6064826E+00	2.5180558E+00
9.6064815E-03	5.0849152E-03	7.5116401E-03	1.7847512E+01	2.6365059E+01	1.2398976E+00	0.0000000E+00
1.0347222E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	8.6345790E-01	0.0000000E+00
1.1087963E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	6.7526398E-01	0.0000000E+00
1.2569444E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	5.2902924E-01	0.0000000E+00
1.5532407E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	4.4229072E-01	0.0000000E+00
1.8495370E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	4.1056063E-01	0.0000000E+00
2.1458333E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.9635683E-01	0.0000000E+00
2.4421296E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.8852480E-01	0.0000000E+00
3.0347222E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.8015191E-01	0.0000000E+00
4.2199074E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.7024667E-01	0.0000000E+00
6.5902778E-02	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.5708514E-01	0.0000000E+00
1.1331019E-01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.3853185E-01	0.0000000E+00
2.0812500E-01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	3.1175344E-01	0.0000000E+00
3.9775463E-01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	2.7317260E-01	0.0000000E+00
4.9256944E-01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	2.5650108E-01	0.0000000E+00
6.8219907E-01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	2.2990385E-01	0.0000000E+00
8.7182870E-01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	2.0848257E-01	0.0000000E+00
1.2510880E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	1.7875058E-01	0.0000000E+00
2.0096065E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	1.4618629E-01	0.0000000E+00
2.1992361E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	1.3924375E-01	0.0000000E+00
2.3888657E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	1.3322357E-01	0.0000000E+00
2.7681250E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	1.2381491E-01	0.0000000E+00
3.5266435E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	1.1131271E-01	0.0000000E+00
5.0436806E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	9.7457544E-02	0.0000000E+00
6.5607176E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	8.8682927E-02	0.0000000E+00
8.0777546E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	8.2470903E-02	0.0000000E+00
9.5947917E+00	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	7.7787252E-02	0.0000000E+00
1.0000000E+01	0.0000000E+00	0.0000000E+00	1.7847512E+01	2.6365059E+01	7.6596993E-02	0.0000000E+00