SIMPLIFIED PREDICTIVE MODELS FOR CO_2 SEQUESTRATION PERFORMANCE ASSESSMENT

RESEARCH TOPICAL REPORT ON TASK #4 REDUCED-ORDER METHOD (ROM) BASED MODELS

Reporting Period: October 1, 2012 through June 30, 2015

Principal Investigator: Dr. Srikanta Mishra
mishras@battelle.org 614-424-5712
Principal Authors: Larry Zhaoyang Jin, Jincong He and Prof. Louis J. Durlofsky
(Author affiliation: Stanford University, Stanford, CA 94305)

Date Report Issued: June 2015

U.S. Department of Energy National Energy Technology Laboratory
DOE Award No. DE-FE0009051, Task #4

Submitting Organization:

Battelle Memorial Institute
505 King Avenue
Columbus, OH 43201
DUNS Number: 00 790 1598

Abstract

Reduced-order models provide a means for greatly accelerating the detailed simulations that will be required to manage CO_2 storage operations. In this work, we investigate the use of one such method, POD-TPWL, which has previously been shown to be effective in oil reservoir simulation problems. This method combines trajectory piecewise linearization (TPWL), in which the solution to a new (test) problem is represented through a linearization around the solution to a previously-simulated (training) problem, with proper orthogonal decomposition (POD), which enables solution states to be expressed in terms of a relatively small number of parameters. We describe the application of POD-TPWL for CO_2-water systems simulated using a compositional procedure. Stanford's Automatic Differentiation-based General Purpose Research Simulator (AD-GPRS) performs the full-order training simulations and provides the output (derivative matrices and system states) required by the POD-TPWL method. A new POD-TPWL capability introduced in this work is the use of horizontal injection wells that operate under rate (rather than bottom-hole pressure) control. Simulation results are presented for CO_2 injection into a synthetic aquifer and into a simplified model of the Mount Simon formation. Test cases involve the use of time-varying well controls that differ from those used in training runs. Results of reasonable accuracy are consistently achieved for relevant well quantities. Runtime speedups of around a factor of 370 relative to full-order AD-GPRS simulations are achieved, though the preprocessing needed for POD-TPWL model construction corresponds to the computational requirements for about 2.3 full-order simulation runs. A preliminary treatment for POD-TPWL modeling in which test cases differ from training runs in terms of geological parameters (rather than well controls) is also presented. Results in this case involve only small differences between training and test runs, though they do demonstrate that the approach is able to capture basic solution trends. The impact of some of the detailed numerical treatments within the POD-TPWL formulation is considered in an Appendix.

Contents

Abstract ii

List of Figures vi

List of Tables vii

Executive Summary ix

1 Introduction 1

2 POD-TPWL for CO_2-Water Systems 4
- 2.1 CO_2-Water Flow Equations . . . 4
- 2.2 POD-TPWL Formulation . . . 6
  - 2.2.1 POD and Constraint Reduction . . . 7
  - 2.2.2 POD-TPWL Point Selection . . . 8
  - 2.2.3 POD-TPWL Workflow . . . 9
  - 2.2.4 Rate-Controlled Wells . . . 9

3 Numerical Simulation Results 12
- 3.1 Model 1: Synthetic Aquifer . . . 12
  - 3.1.1 Problem Set Up . . . 12
  - 3.1.2 POD-TPWL Results with BHP Controls (Model 1) . . . 14
  - 3.1.3 POD-TPWL Results with Rate Controls (Model 1) . . . 17
- 3.2 Model 2: Mount Simon Formation . . . 21
  - 3.2.1 Problem Set Up . . . 21
  - 3.2.2 POD-TPWL Results with BHP Controls (Model 2) . . . 22
  - 3.2.3 POD-TPWL Results with Rate Controls (Model 2) . . . 23
- 3.3 Summary . . . 30

4 Geological Perturbation 30
- 4.1 POD-TPWL Formulation . . . 30
- 4.2 Problem Set Up . . . 31
- 4.3 POD-TPWL Results . . . 33
- 4.4 Summary . . . 36

5 Summary and Conclusions 36

Acknowledgments 38

Appendix A Constraint Reduction for POD-TPWL 39
- A.1 Summary of Appendix A . . . 39
- A.2 Introduction . . . 39
- A.3 Problem Description . . . 42
- A.4 POD-TPWL Model and Assessment of Error . . . 44
  - A.4.1 Trajectory Piecewise Linearization . . . 45
  - A.4.2 Proper Orthogonal Decomposition . . . 46
  - A.4.3 Constraint Reduction . . . 47
  - A.4.4 Error Propagation . . . 48

A.4.5 Total Error of POD-TPWL Model 49
A.5 Optimal Constraint Reduction Procedures 50
A.5.1 General Development 50
A.5.2 Galerkin Projection 51
A.5.3 Petrov-Galerkin Projection 52
A.6 Stability Criteria 53
A.7 Numerical Implementation and Results 55
A.7.1 POD-TPWL Implementation 56
A.7.2 Error Definitions 57
A.7.3 Case 1: Oil-Water Flow with Equal Phase Densities 58
A.7.4 Case 2: Oil-Water Flow with Unequal Phase Densities 61
A.7.5 Case 3: Compositional Simulation 66
A.8 Inverse Projection and Weighted Inverse Projection Constraint Reduction Methods . 69
A.8.1 Method Development 71
A.8.2 Numerical Results using IP and WIP 72
A.9 Concluding Remarks 74

Appendix B POD-TPWL for CO_2 EOR **76**

References **88**

List of Figures

1 Simulation grid (left, areal view) and horizontal injection wells (right) 13
2 Permeability field for storage aquifer ($\log k$ is shown) 13
3 Time-varying BHPs for training and target simulations (Model 1) 15
4 CO_2 injection rates for test case with $\alpha = 0.3$ (Model 1) 15
5 CO_2 injection rates for test case with $\alpha = 0.5$ (Model 1) 16
6 CO_2 injection rates for test case with $\alpha = 0.8$ (Model 1) 16
7 Time-varying rate specifications for training and target simulations (Model 1) 17
8 CO_2 injection well BHPs for test case with $\alpha = 0.5$ (Model 1) 18
9 CO_2 injection well BHPs for test case with $\alpha = 1.0$ (Model 1) 18
10 Color maps for CO_2 overall molar fraction at 4000 days with $\alpha = 1.0$ (Model 1) . . . 19
11 Comparison of CO_2 overall molar fraction between POD-TPWL and AD-GPRS at 4000 days and 10,000 days for test case with $\alpha = 1.0$ (Model 1) 20
12 Areal grid and well locations for simplified Mount Simon model 21
13 Permeability field for simplified Mount Simon model ($\log k_x$ is shown, k_x in mD) . . 22
14 Time-varying BHPs for training and target simulations (Model 2) 22
15 CO_2 injection rates for test case with $\alpha = 0.3$ (Model 2) 23
16 CO_2 injection rates for test case with $\alpha = 0.8$ (Model 2) 24
17 Time-varying rate specifications for training and target simulations (Model 2) 25
18 CO_2 injection well BHPs for test case with $\alpha = 1.0$ (Model 2) 26
19 Color maps for CO_2 overall molar fraction z_g in layer 25 (Model 2) 27
20 Absolute differences in z_g in layer 25 at 4000 and 6000 days (Model 2). Upper plots display differences between full-order training and test results, while lower plots display differences between POD-TPWL and AD-GPRS test solutions 28
21 Comparison of pressure and CO_2 molar fraction between POD-TPWL and full-order reference solutions (Model 2) 29
22 Permeability field (in $\log k$) and well locations for geological perturbation example . 32
23 Injection rates in training and test cases (rates are the same for both wells) 32
24 Injection well BHPs for test cases with all transmissibilities perturbed by constant factors (Well 1) 34
25 Injection well BHPs for test case with perturbed vertical transmissibilities (Well 1) . 35
26 Injection well BHPs for test case with perturbed transmissibilities in layers 23–27 (Well 1) 35
27 Reservoir model for Cases 1 and 2 (permeability in the x-direction is shown) 58
28 Time-varying BHPs for the primary training simulation for Cases 1 and 2 59
29 Time-varying BHPs for the test simulation for Cases 1 and 2 59
30 Amplification factor γ^i for each time step in Case 1 60
31 Oil production rate for Producer 1 in Case 1. Results for Test (AD-GPRS), GLK_70_100 and PG_70_100 essentially overlay one another 61
32 Water injection rate for Injector 1 in Case 1. Results for Test (AD-GPRS) and GLK_70_100 essentially overlay one another 62
33 Oil production rate for Producer 1 in Case 2 63
34 Water injection rate for Injector 1 in Case 2 63
35 Amplification factor γ^i for each time step in Case 2 64
36 Maps of $\log_{10}(\max_i \gamma^i)$ for Cases 1 and 2 65
37 Water injection rate for Injector 1 in Case 2, with l_p and l_S selected based on Figure 36c 66
38 Reservoir model for Case 3 (log-permeability is shown) 67

39 Time-varying BHPs for the primary training simulation for Case 3 67
40 Time-varying BHPs for the test simulation for Case 3 68
41 Maps of $\log_{10}(\max_i \gamma^i)$ for Case 3 68
42 Oil production rate for Producer 1 in Case 3 69
43 Gas production rate for Producer 1 in Case 3 70
44 Gas injection rate for Injector 1 in Case 3 70
45 Maps of $\log_{10}(\max_i \gamma^i)$ for IP and WIP for Case 3 72
46 Geological model and well locations (from [20, 39]) 76
47 Time-varying BHPs for training case 1 77
48 Time-varying BHPs for training case 2 77
49 Time-varying BHPs for test case 78
50 Oil production rates 79
51 Gas production rates 80
52 CO_2 injection rates 81

List of Tables

1 Timings for various modeling components (in seconds) 37
2 Summary of error for Case 1 61
3 Summary of error for Case 2 66
4 Summary of error for Case 1, with IP and WIP 73
5 Summary of error for Case 2, with IP and WIP 73
6 Summary of error for Case 3, with IP and WIP 73

Executive Summary

The methods and results presented in this topical report represent the accomplishments under Task 4 of the overall project on 'Simplified Predictive Models for CO_2 Sequestration Performance Assessment.' Task 4 was concerned with Reduced-Order Method (ROM) based Models, and the research associated with this task was performed at Stanford University. The need for reduced-order modeling is motivated by the observation that, although flow simulation can be used to design and manage CO_2 sequestration projects, the large number of detailed runs required for some applications (such as computational optimization and uncertainty assessment) can lead to great computational expense. Computationally-efficient procedures, including numerical reduced-order models, which have been applied in related areas such as oil reservoir simulation, may thus be very useful for these problems.

In this work, we explore the use of trajectory piecewise linearization (TPWL) combined with proper orthogonal decomposition (POD) for simulating CO_2 storage problems. POD-TPWL models of this type have been successfully used for oil-water and oil-gas compositional reservoir simulation problems. The basic approach with POD-TPWL is to first perform one (or a few) full-order 'training' runs, which entail high-fidelity (full-order) flow simulations under a prescribed set of well controls (e.g., time-varying bottom-hole pressures or rates). For subsequent (test) runs, which involve different well control settings, the solution at each time step is represented based on a linearization around a training solution. The use of POD, which allows us to represent solution states (e.g., pressure and overall mole fraction in every grid block) in terms of a small number of parameters, along with a constraint reduction procedure, which projects the set of governing equations into a low-dimensional subspace, provides a high degree of efficiency.

The full-order simulations applied in this work use a two-phase, two-component (CO_2 and water) formulation within Stanford's Automatic Differentiation-based General Purpose Research Simulator (AD-GPRS). This simulator was modified to output the state and derivative matrices required to construct the POD-TPWL model. New features introduced in this work, in addition to the application of POD-TPWL to CO_2 sequestration simulations, are the use of rate-control specifications for wells and the incorporation of horizontal injectors into the model. Because of the way in which AD-GPRS represents wells, the use of rate-controlled wells in POD-TPWL requires additional matrix manipulations in the model construction step.

CO_2 storage with both a synthetic (channelized) aquifer and an approximate model of the Mount Simon formation (planned for use with FutureGen 2.0) is considered for test cases that involve wells controlled by both time-varying bottom-hole pressures and rates. Generally accurate results are obtained for well quantities and for CO_2 plume location, though the accuracy of the POD-TPWL model is seen to degrade as the controls used in test cases deviate from those applied in training runs. Runtime speedups with POD-TPWL for these cases are about a factor of 370 relative to high-fidelity AD-GPRS simulations. The overhead required to construct the POD-TPWL model (including training runs) is equivalent to about the time required for 2.3 full-order runs.

The POD-TPWL model is then extended to allow parameters associated with the geologic model to be perturbed in test runs. Preliminary results using this capability in two-dimensional models, in which all block-to-block transmissibilities are multiplied by a constant value relative to the training run, demonstrate that the POD-TPWL model is able to capture general trends in the relevant well quantities. The differences between test- and training-case results are, however, very small in the scenarios considered. Results are also presented for a CO_2-enhanced oil recovery problem, which demonstrates the use of POD-TPWL for problems where CO_2 is both utilized and sequestered.

An Appendix to this report presents a detailed assessment of constraint reduction procedures

for POD-TPWL models of the type considered here. As noted above, the constraint reduction procedure projects the set of governing equations into an appropriate subspace of much lower dimension. The approach used in previous POD-TPWL models of oil-water systems was the Galerkin projection procedure, in which the left-projection matrix is the transpose of the POD basis matrix used to concisely represent the system states. In this work, we show that the use of a (different) so-called Petrov-Galerkin procedure leads to much better stability properties in POD-TPWL models of oil-water and oil-gas compositional systems. This is the approach used in all of the CO_2 storage simulations presented in this report.

1 Introduction

The carbon dioxide generated by the combustion of fossil fuels in power stations represents a large component of worldwide greenhouse gas emissions. These emissions are major contributors to global climate change, so it is important that they be reduced substantially. The use of carbon capture and storage (CCS), in which CO_2 is separated from the flue gas stream exiting the power plant and then injected into deep subsurface formations, represents a means for reducing CO_2 emissions to the atmosphere [8]. There are two large CCS projects that have been operating for many years, namely the Sleipner project in the Norwegian North Sea [2], started in 1996, and the Weyburn project in Canada [73], started in 2000. Another large CO_2 storage project, the In Salah project in Algeria [76], started in 2004 but suspended CO_2 injection in 2011. Projects of similar scale have also been proposed in the United States, such as that associated with (recently canceled) FutureGen 2.0 [7].

Large-scale subsurface flow simulation can be used to design and manage CO_2 storage projects, and a number of modeling tools have been introduced; e.g., [26, 29, 30, 57, 58]. Formations under consideration for use in CO_2 storage include depleted oil and gas reservoirs and deep saline aquifers (the latter are much more widely available). Issues of concern in carbon storage include those associated with injectivity and pressure buildup (i.e., can sufficient CO_2 be injected without activating faults or fracturing the cap rock), tracking the location of the CO_2 plume, and minimizing the possibility for leakage of CO_2 or brine into fresh-water aquifers (or into the atmosphere).

Several researchers have applied computational optimization procedures in order to minimize the risk of leakage from the storage formation. This includes the work of Nghiem et al. [50, 51], who optimized injection strategies for individual wells, and studies by Cameron & Durlofsky [13, 14], who optimized well locations and time-varying injection rates for multiple horizontal injectors, with the goal of minimizing the amount (or the overall mobility) of CO_2 at the top of the formation. In the latter study [14], the impact of geological uncertainty was considered, both in the placement of injection wells and in their subsequent operation. History matching was also performed, in conjunction with optimization, within a 'closed-loop' framework.

Studies of this type are limited, however, by the need to perform (potentially) thousands of simulation runs. In many optimization problems, the required simulations often resemble one another fairly closely, since well control parameters are varied gradually during the course of the optimization. This suggests that computationally-efficient alternative procedures, such as reduced-order models (ROMs) or other types of 'proxy' or 'surrogate' models, may be appropriate for these problems. Numerical ROM procedures have been widely investigated within the context of oil reservoir simulation (as discussed below), though their use for CO_2 storage problems has not yet been considered. The goal of this work is to test the application of one such ROM, namely POD-TPWL (proper orthogonal decomposition — trajectory piecewise linearization), for simulations of CO_2 storage.

Although numerical ROMs of the type considered here do not appear to have been applied for CO_2 storage problems, a number of other proxy models, which use simplified physics and

statistical fits, have been developed. Burton et al. [11], for example, implemented a simplified-physics model for CO_2 injection based on modified Buckley-Leverett theory [52], which provided reasonable accuracy in gas saturation distribution. Oruganti & Mishra [56] evaluated this model by comparing its performance with results from a numerical simulator, and this led them to develop a modified version of the model with improved accuracy. All of these models, however, are for a single well injecting into a homogeneous formation, so their direct use for, e.g., rate optimization in multiwell problems, will be limited.

Simplified-physics models have also been developed by Nordbotten et al. [53–55] and Gasda et al. [33]. These researchers used vertical equilibrium assumptions to construct simplified analytical, numerical and numerical-analytical hybrid models, which are able to simulate CO_2 migration at reduced computational cost. These models only considered vertical wells, however, and it is not clear if and when they are applicable (or what modifications are required) for horizontal injectors. In addition, these approaches approximate some important physical effects, such as the dissolution of CO_2 into brine and saturation variations within the brine phase. In many cases these are reasonable approximations, but in some situations they may lead to inaccuracy.

Wriedt et al. [75] developed a response surface methodology using a Box-Behnken experimental design to quantify model response for CO_2 injection problems. Anbar [1] proposed an approach with a design based on space-filling maxmin Latin Hypercube sampling to provide estimates of the CO_2 storage capacity of an aquifer. Schuetter et al. [64, 65] tested and analyzed these statistical proxy models, and observed that they were able to provide acceptable results in the cases tested. However, these approaches can lose accuracy when the response surfaces are very smooth (in which case the output is not sensitive to the parameters), or when a test case falls at or near the 'edge' of the input space. In general, to assure a high degree of accuracy in output quantities, a large number of samples are required (which means a large number of full-order simulations must be performed).

In our work here, we consider ROMs in which the proxy model is constructed by applying a set of specific numerical procedures to the full system of discretized equations. Such approaches commonly employ proper orthogonal decomposition (POD), which enables the representation of the full-order states (e.g., pressure and saturation in every grid block in a two-phase flow problem) in a low-dimensional subspace. A number of ROM procedures employ only POD (e.g., [18, 43, 67, 70]), though these approaches have been found to provide limited speedup (e.g., a factor of ten at best) for nonlinear problems of the type considered here.

Nonlinearity can be treated efficiently through the use of linearization (discussed below) or by introducing an approximation of the nonlinear terms. The latter approach includes the discrete empirical interpolation method (DEIM), which entails the interpolation of nonlinear terms following their evaluation at a few selected locations. DEIM has been applied in the context of porous media problems by Chaturantabut & Sorensen [22] and Ghasemi et al. [36]. These approaches are quite promising, though so far they have only been applied to porous media problems with relatively simple physics. In addition, DEIM techniques are more invasive with respect to the simulator than

the POD-TPWL procedure developed here.

POD-TPWL methods handle nonlinear effects through (piecewise) linearization. With this approach, new solutions are represented using a truncated Taylor series expansion around previously simulated solutions. The solution states and the system of equations are projected into a low-dimensional subspace using POD-based procedures. The method requires overhead corresponding to the time required for a few full-order simulations (most of this time is spent performing full-order 'training' runs), though subsequent (test) runs are extremely fast. The basic TPWL approach was introduced by Rewienski and White [62] for the modeling of nonlinear circuits and micromachined devices. Within a reservoir simulation setting, POD-TPWL has been applied to oil-water problems by Cardoso & Durlofsky [16, 17], He et al. [41] and Fragoso et al. [31], to idealized thermal problems by Rousset et al. [63], and to oil-gas compositional problems by He & Durlofsky [39, 40]. Reasonable levels of accuracy have been achieved for simulations in which the well controls in test runs are relatively 'close' to those used in training runs. Runtime speedups of up to a factor of ~ 500 have been reported [16]. The method has also been successfully applied for well control optimization [16, 17, 31, 39], though some 'retraining' may be required when the test controls differ substantially from those used in the initial training runs.

The surrogate-modeling approaches discussed above are applicable in certain cases, but all have limitations which restrict their use. For example, as noted earlier, the simplified-physics models in [11] and [56] were for single wells operating in homogeneous formations. It may be challenging to generalize these approaches to heterogeneous models with multiple wells. Statistical proxy models also have important limitations, including the need for large numbers of 'training' runs (samples) in problems that are characterized by many parameters (especially when detailed output information is desired). Within the context of numerical ROMs, neither POD-DEIM nor POD-TPWL procedures have been developed for carbon storage applications. POD-TPWL has, however, been successfully applied for compositional reservoir simulation. These models have not included horizontal or rate-controlled wells, and both of these capabilities will be important for realistic carbon storage simulations.

Our goal in this study is to develop and apply a POD-TPWL procedure for geological carbon sequestration problems. We will consider CO_2-water compositional systems, with injection accomplished via multiple horizontal wells, which could be controlled by specifying either bottom-hole pressure or rate. This capability will be tested using both a synthetic aquifer model and an approximate model for the Mount Simon formation (which was the injection target for FutureGen 2.0). Both models involve nonuniform grids covering very large domains (to represent regional aquifers). We will also implement a prototype approach for treating geological perturbations within the POD-TPWL framework.

This report proceeds as follows. In Section 2, we present the governing equations for the flow of CO_2 and water in subsurface formations. The POD-TPWL formulation for this system is then described. In Section 3, POD-TPWL results are presented for both a synthetic model with two horizontal CO_2 injectors, and for an idealized Mount Simon model involving four horizontal CO_2

injectors. The formulation and initial results for POD-TPWL models with geological perturbations are presented in Section 4. In Section 5 we provide concluding remarks and some suggestions for future research.

Appendix A presents a detailed assessment of constraint reduction procedures within the context of POD-TPWL models for oil-gas and oil-water systems. This Appendix will appear as a paper in *International Journal for Numerical Methods in Engineering*. Within the body of this report, this work is referenced as He and Durlofsky [40]. Results demonstrating the use of POD-TPWL for CO_2 EOR (enhanced oil recovery), which represents a potential means to simultaneously utilize and store anthropogenic CO_2, are presented in Appendix B.

2 POD-TPWL for CO_2-Water Systems

In this section, we present the full-order compositional model for the CO_2-water problem, including the choice of primary variables. The formulation and workflow for the CO_2-water POD-TPWL model are then described. The formulation and presentation here closely resemble those for the oil-gas compositional POD-TPWL implementation presented by He and Durlofsky [39, 40], and those papers should be consulted for full details.

2.1 CO_2-Water Flow Equations

The CO_2-water system considered in this work could be treated using either a black-oil or a compositional formulation. Here we use a compositional approach. We proceed in this way because the compositional model treats the solubility of CO_2 as a function of both pressure and composition in the water phase, rather than as a function of pressure only, as in the black-oil model implemented in Stanford's AD-GPRS (which is the simulator used in this work). This treatment provides a more accurate representation of dissolution trapping of CO_2, which is an important trapping mechanism.

The system contains two components – CO_2, designated g (for gas), and water, designated w. Both components can exist in either the gas or water phases, which are also denoted by g and w. The mass balance equation for each component c ($c = g, w$) can be written as:

$$\begin{aligned}\frac{\partial}{\partial t}[\phi(S_w\rho_w x_c + S_g\rho_g y_c)] \; - \; & \nabla\cdot[\mathbf{k}(\lambda_w\rho_w x_c\nabla\Phi_w + \lambda_g\rho_g y_c\nabla\Phi_g)] \\ & + \; \sum_w(\rho_w x_c q_w^w + \rho_g y_c q_g^w) = 0.\end{aligned} \tag{1}$$

Here t denotes time, ϕ is porosity, S_g and S_w indicate gas and water phase saturation, and x_c and y_c represent the molar fractions of component c in the water and gas phases. Other variables include the permeability tensor $\mathbf{k}$, phase mobility λ_j (where $\lambda_j = k_{rj}/\mu_j$, with $j = g$ or w, k_{rj} denotes relative permeability to phase j, and μ_j is phase viscosity), and phase density ρ_j. The variable q_j^w denotes the source term for well w. The phase potential Φ_j is defined as

$$\Phi_j = p_j - \rho_j g(D - D_{\text{ref}}), \tag{2}$$

where p_j is phase pressure, D is depth, D_{ref} is a reference depth, and g is gravitational acceleration.

The compositional system described by Eq. 1 is fully defined after the introduction of phase equilibrium and other (simple) constraints (e.g., $S_g + S_w = 1$). See [38] for details. Note that, in this study, we take the capillary pressure between the gas and water phases to be zero. This simplification may be reasonable for injection-period simulations of the type performed here, but capillary pressure effects (particularly capillary heterogeneity) have been shown to be important during the equilibration phase [45]. These effects should be included in future implementations.

As described in [3], the full system (in general) involves $n_c n_p + 2n_p$ equations and variables for each grid block, where n_c and n_p denote the number of components and phases. In our CO_2-water model, $n_c = n_p = 2$. We thus have a total of eight unknowns in the general case (which are p_g, p_w, S_g, S_w and x_c and y_c in both phases, for $c = g, w$). Neglecting capillary pressure, this reduces to seven unknowns. The system is, however, fully defined by only two variables (the so-called primary variables), which are determined by solving Eq. 1 for both components. Various choices can be used for the primary variables, as discussed in [39]. Consistent with that work, the primary variables used here are pressure and overall molar fraction for the water component, denoted z_w and given by $z_w = S_g y_w + S_w x_w$. This formulation is referred to as the molar formulation, which is discussed in detail by [72].

As in previous POD-TPWL implementations for reservoir simulation [16, 39, 41, 63], the full-order simulation equations are discretized using the usual finite volume formulation with a fully-implicit time discretization. Following discretization, the set of nonlinear algebraic equations can be expressed as:

$$\mathbf{g}(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^{n+1}) = \mathbf{0}, \tag{3}$$

where $\mathbf{g}$ is the residual vector we seek to drive to zero, $\mathbf{x}$ represents the system states (p and z_w in every grid block), n indicates the previous time level and $n+1$ the next time level, and $\mathbf{u}^{n+1}$ designates the control parameters, which define the well (source) terms. In an actual simulation, $\mathbf{x}^n$ is known and $\mathbf{u}^{n+1}$ is specified, and the goal is to compute $\mathbf{x}^{n+1}$. In previous POD-TPWL implementations, wells were controlled by specifying bottom-hole pressure (BHP). In this work, we introduce the use of well rate specifications, which is useful for CO_2 storage problems since we often wish to inject a particular volume of CO_2 at each time step.

The nonlinear system defined by Eq. 3 is typically solved using Newton's method. This entails, at each iteration for every time step, constructing the (sparse) Jacobian matrix, which is of dimensions $2n_b \times 2n_b$, and then solving a linear system of dimension $2n_b$, where n_b is the number of grid blocks in the model (the factor of two enters since there are two primary equations and unknowns per grid block). In models involving relatively few components (as is the case here), this linear solution is typically the most time consuming part of the flow simulation. The attraction of POD-TPWL is that we avoid the construction and solution of this high-dimensional ($2n_b \times 2n_b$) system during the inline (runtime) computations.

2.2 POD-TPWL Formulation

The POD-TPWL procedure entails linearization of the (discrete) equations, the construction of basis matrices to represent the states concisely, constraint reduction to reduce the number of equations that must be solved, and a point selection scheme to determine the previous state/controls around which to linearize. We now describe each of these treatments. See [39, 40] for full details.

In trajectory piecewise linearization (with or without POD), the idea is to use the states and derivative matrices generated and saved during so-called training runs for the representation of solutions to problems that involve a different set of controls. Simulations with new controls are referred to as test runs. In the equations presented below, the superscripts i and $i+1$ denote consecutive time steps in the training simulations, and n and $n+1$ indicate consecutive time steps in test simulations.

We proceed by representing the residual vector for a test run in terms of a Taylor series expansion around the residual vector for a training run. Applying Eq. 3, we then have

$$\mathbf{g}^{n+1} = \mathbf{0} \approx \mathbf{g}^{i+1} + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}}(\mathbf{x}^{n+1} - \mathbf{x}^{i+1}) + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i}}(\mathbf{x}^{n} - \mathbf{x}^{i}) + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{u}^{i+1}}(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}), \tag{4}$$

where $\mathbf{g}^{i+1} = \mathbf{g}(\mathbf{x}^{i+1}, \mathbf{x}^{i}, \mathbf{u}^{i+1}) = \mathbf{0}$ since this is a (previously simulated) training run solution, and $\mathbf{g}^{n+1} = \mathbf{g}(\mathbf{x}^{n+1}, \mathbf{x}^{n}, \mathbf{u}^{n+1})$ is also set to $\mathbf{0}$ since this is the equation we wish to solve. Note that the Taylor series expansion is truncated at first order.

After rearrangement, Eq. 4 can be written as:

$$\mathbf{J}^{i+1}(\mathbf{x}^{n+1} - \mathbf{x}^{i+1}) = -[\mathbf{A}^{i+1}(\mathbf{x}^{n} - \mathbf{x}^{i}) + \mathbf{B}^{i+1}(\mathbf{u}^{n+1} - \mathbf{u}^{i+1})], \tag{5}$$

with the three matrices defined as

$$\mathbf{J}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}}, \quad \mathbf{A}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i}}, \quad \mathbf{B}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{u}^{i+1}}. \tag{6}$$

Note that $\mathbf{J}^{i+1}$ is the Jacobian matrix at time step $i+1$ of the training simulation, evaluated for the converged system.

In order to achieve a high degree of computational efficiency, we now seek to approximate the $2n_b \times 2n_b$ linear system given by Eq. 5 in a low-dimensional subspace. Two additional steps are required in order to accomplish this – the representation of states using proper orthogonal decomposition, and the application of constraint reduction to project the set of linear equations to a low dimension. We now describe these two procedures.

2.2.1 POD and Constraint Reduction

In proper orthogonal decomposition (POD), a basis matrix $\mathbf{\Phi}$ is introduced to enable the representation of the state variables $\mathbf{x}$ in terms of a reduced state vector $\boldsymbol{\xi}$:

$$\mathbf{x} \approx \mathbf{\Phi}\boldsymbol{\xi}. \tag{7}$$

We construct $\mathbf{\Phi}$ by performing singular value decomposition (SVD) of so-called snapshot matrices formed during training simulations. Following [39], this procedure is performed separately for the two state variables p and z_w. Recall that the full-order training runs provide the full solution at each time step. The states determined during these runs are entered as columns in the snapshot matrices $\mathbf{X}_p$ and $\mathbf{X}_z$; i.e., we construct

$$\mathbf{X}_p = [\mathbf{p}^1, \mathbf{p}^2, \cdots, \mathbf{p}^L], \quad \mathbf{X}_z = [\mathbf{z}_w^1, \mathbf{z}_w^2, \cdots, \mathbf{z}_w^L], \tag{8}$$

where the vectors $\mathbf{p}^i$ and $\mathbf{z}_w^i$ indicate the pressure and overall water molar fraction in each of the n_b grid blocks at time step i in a training run. The total number of snapshots is L; these could all derive from one training run, or they could be generated using multiple training runs. Each of the snapshot matrices is thus of dimensions $n_b \times L$.

From the SVDs of $\mathbf{X}_p$ and $\mathbf{X}_z$ (which are performed separately), we obtain the left singular vectors, which comprise the columns of the basis matrices $\mathbf{\Phi}_p$ and $\mathbf{\Phi}_z$. Each matrix could contain a maximum of L columns, though we typically do not retain all L columns. Rather, based on an energy criterion, or limited numerical experimentation [39, 40], we retain l_p columns in $\mathbf{\Phi}_p$ and l_z columns in $\mathbf{\Phi}_z$ (generally, $l_p \neq l_z$).

Using these matrices, the state variables are represented as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_p \\ \mathbf{x}_z \end{bmatrix} \approx \mathbf{\Phi}\boldsymbol{\xi} = \begin{bmatrix} \mathbf{\Phi}_p & 0 \\ 0 & \mathbf{\Phi}_z \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi}_p \\ \boldsymbol{\xi}_z \end{bmatrix}, \tag{9}$$

where $\boldsymbol{\xi}_p$ and $\boldsymbol{\xi}_z$ are the reduced variables for pressure and overall water molar fraction. Defining $l = l_p + l_z$, this projection reduces the number of primary unknowns from $2n_b$ to l, with $l \ll 2n_b$. Note that the entries in the overall $\mathbf{\Phi}$ matrix are ordered as shown in Eq. 9 if we order the unknowns with pressure in all blocks appearing first, followed by molar fraction of water in all blocks (i.e., $p_1, p_2, \ldots, p_{n_b}, z_{w,1}, z_{w,2}, \ldots, z_{w,n_b}$). If we order the unknowns as $(p_1, z_{w,1})$, $(p_2, z_{w,2})$, $\ldots$, (p_{n_b}, z_{w,n_b}), then the entries in $\mathbf{\Phi}_p$ and $\mathbf{\Phi}_z$ are interspersed within the overall $\mathbf{\Phi}$ matrix. See [18] for the detailed structure of $\mathbf{\Phi}$ with this ordering.

Inserting the representation for the states in Eq. 9 into (linearized) Eq. 5 leads to a system of $2n_b$ equations in l variables. In order to reduce the number of equations to l (to render the system well posed), we premultiply the resulting set of equations by a matrix $\mathbf{\Psi}^T$, where superscript T denotes transpose. The matrix $\mathbf{\Psi}$, which is referred to as the constraint reduction matrix, is of

dimensions $2n_b \times l$. The low-order linearized equation is now given by:

$$\mathbf{\Psi}^T \mathbf{J}^{i+1} \mathbf{\Phi} (\boldsymbol{\xi}^{n+1} - \boldsymbol{\xi}^{i+1}) = -\mathbf{\Psi}^T [\mathbf{A}^{i+1} \mathbf{\Phi} (\boldsymbol{\xi}^n - \boldsymbol{\xi}^i) + \mathbf{B}^{i+1} (\mathbf{u}^{n+1} - \mathbf{u}^{i+1})], \tag{10}$$

which involves l equations and l unknowns. This equation can be expressed as

$$\boldsymbol{\xi}^{n+1} = \boldsymbol{\xi}^{i+1} - (\mathbf{J}_r^{i+1})^{-1} [\mathbf{A}_r^{i+1} (\boldsymbol{\xi}^n - \boldsymbol{\xi}^i) + \mathbf{B}_r^{i+1} (\mathbf{u}^{n+1} - \mathbf{u}^{i+1})], \tag{11}$$

where the reduced derivative matrices are defined as

$$\mathbf{J}_r^{i+1} = (\mathbf{\Psi}^{i+1})^T \mathbf{J}^{i+1} \mathbf{\Phi}, \quad \mathbf{A}_r^{i+1} = (\mathbf{\Psi}^{i+1})^T \mathbf{A}^{i+1} \mathbf{\Phi}, \quad \mathbf{B}_r^{i+1} = (\mathbf{\Psi}^{i+1})^T \mathbf{B}^{i+1}. \tag{12}$$

Here, $\mathbf{J}_r^{i+1}$ and $\mathbf{A}_r^{i+1}$ are both of dimensions $l \times l$, and $\mathbf{B}_r^{i+1}$ is of dimensions $l \times n_u$, where n_u is the number of control variables. For typical well control problems, there are many more grid blocks than wells, so $n_u \ll n_b$.

The choice of constraint reduction matrix is considered in detail in [40] (Appendix A). There it was shown that, for some types of problems, the commonly applied Galerkin projection procedure (in which $\mathbf{\Psi}^{i+1} = \mathbf{\Phi}$, where $\mathbf{\Phi}$ is the basis matrix for the state variables) can lead to numerical instability in the POD-TPWL model. Rather than use Galerkin projection, He and Durlofsky [39, 40] suggested the use of a Petrov-Galerkin procedure in which $\mathbf{\Psi}^{i+1} = \mathbf{J}^{i+1} \mathbf{\Phi}$. This approach, motivated by results presented in [19], was shown to lead to much more stable POD-TPWL models. It is important to note, however, that numerical stability is still not guaranteed with this procedure.

As noted above, one or more training runs can be used to provide snapshots for the construction of the POD basis. Following [39], in this work we typically use two training runs for snapshot generation. However, we save and reduce the Jacobian (and other derivative) matrices from only one of these training simulations. This run is referred to as the primary training run. By using this procedure, we save on preprocessing computation and storage.

2.2.2 POD-TPWL Point Selection

At each time step in a POD-TPWL run, the sequential training states i and $i+1$ around which we linearize must be determined (see Eqs. 11 and 12). We refer to this determination as point selection (i.e., we select the training point around which the linearization is performed). This linearization point is found by minimizing a measure of distance between the current (reduced) test-run state and all points in the primary training run.

Following [39], we define this distance ($d^{n,j}$) as

$$d^{n,j} = d_z^{n,j} + \gamma d_T^{n,j}. \tag{13}$$

Here $d_z^{n,j}$ and $d_T^{n,j}$ denote the relative difference in (reduced) mole fraction and in estimated pore volume injected (PVI), respectively, and γ is a weighting parameter (we take $\gamma = 10$). The specific

definitions for these two contributions are

$$d_z^{n,j} = \frac{|\boldsymbol{\xi}_z^n - \boldsymbol{\xi}_z^j|}{|\boldsymbol{\xi}_z^n| + \epsilon}, \quad d_T^{n,j} = \frac{|\int_0^{t^n} q^n \mathrm{d}t - \int_0^{t^n} q^j \mathrm{d}t|}{\int_0^{t^n} q^n \mathrm{d}t + \epsilon}, \tag{14}$$

where $\boldsymbol{\xi}_z^n$ and $\boldsymbol{\xi}_z^j$ are reduced overall molar fraction (of water) for the test run at time step n and for the saved point j in the primary training run. The variables q^n and q^j designate the total injection rate (over all wells) at time step n in the test run and at time step j in the training run. The ϵ term, set as 0.01, is relevant only at very early times. By finding the value of j that minimizes $d^{n,j}$ in Eq. 13, we find the state (and its associated derivative matrices) in the primary training run that is closest to the current test state in terms of $\boldsymbol{\xi}_z$ and PVI. Eq. 14 is directly applicable when rates are specified in the test runs. If BHPs are specified, rates must be computed from the well model. The way in which this is accomplished is discussed later (see Eq. 16 below).

2.2.3 POD-TPWL Workflow

Prior to using POD-TPWL to simulate test cases, the reduced-order model must be constructed from training run results. This entails preprocessing (offline) computations, which include training runs, construction of the basis matrices $\boldsymbol{\Phi}_p$ and $\boldsymbol{\Phi}_z$, and construction of the reduced derivative matrices appearing in Eqs. 11 and 12. Stanford's Automatic Differentiation-based General Purpose Research Simulator, AD-GPRS [79], has been modified to provide the necessary derivative information. For the cases considered here, the additional computation required to construct the POD-TPWL model can entail nearly 1/3 of the time required for a full-order training run. Thus, assuming two training runs are performed, the overall preprocessing POD-TPWL overhead corresponds to about the time for 2.3 full-order simulations. For the CO_2 storage simulations considered here, runtime speedups using POD-TPWL of nearly a factor of 400 are observed. Thus, if the model is to be run many times, the POD-TPWL procedure can be very cost effective. If, however, only a few runs are required, it is more efficient to simply run the full-order model.

During POD-TPWL runs (i.e., inline computations), at each time step, the training states i and $i+1$ are first found (by minimizing $d^{n,j}$), after which Eq. 11 is solved to determine $\boldsymbol{\xi}^{n+1}$. If actual states are required, these can be constructed by applying $\mathbf{x}^{n+1} \approx \boldsymbol{\Phi}\boldsymbol{\xi}^{n+1}$. Typically, states are only required at well blocks, where they are used to compute flow rates. As discussed by [39], a flash must be performed to construct secondary variables such as saturation. See [39] for a flow chart of the offline and inline computations and for additional implementation details.

2.2.4 Rate-Controlled Wells

New features introduced in this work include the use of horizontal wells and the control of wells through rate (rather than BHP) specifications. Handling horizontal wells is relatively straightforward and mostly involves input and output specifications. The use of well rate as a control variable is more complicated, as described below.

Prior to discussing rate-controlled wells, it is useful to first provide some details regarding the use of BHPs as the control parameters, since AD-GPRS essentially models wells in both cases using BHPs, which are then related to rates. For BHP-controlled wells, the derivative matrices needed to construct the POD-TPWL model are $\partial \mathbf{g}^{i+1}/\partial \mathbf{x}^{i+1}$ ($\mathbf{J}^{i+1}$) and $\partial \mathbf{g}^{i+1}/\partial \mathbf{u}^{i+1}$ ($\mathbf{B}^{i+1}$), as is evident from Eqs. 5 and 6. Note that the matrix $\partial \mathbf{g}^{i+1}/\partial \mathbf{x}^{i}$ ($\mathbf{A}^{i+1}$) does not enter this discussion because it is not affected by changing well control from BHPs to rates. It is convenient to define an 'extended' Jacobian matrix $\mathbf{J}_{\text{ext}}$ containing both $\mathbf{J}$ and $\mathbf{B}$:

$$\mathbf{J}_{\text{ext}} = \begin{bmatrix} \mathbf{J}_{RR} & \mathbf{J}_{RW} \end{bmatrix} = \begin{bmatrix} \mathbf{J} & \mathbf{B} \end{bmatrix}, \tag{15}$$

where we have dropped superscript $i+1$ for simplicity. Here the subscript on $\mathbf{J}_{RR}$ (which is of dimensions $2n_b \times 2n_b$) indicates that it is the derivative of the reservoir equations with respect to primary reservoir variables, and that on $\mathbf{J}_{RW}$ (of dimensions $2n_b \times n_w$, where n_w is the number of wells) indicates that it is the derivative of the reservoir equations with respect to well BHPs. These matrices are precisely as defined in Eq. 6.

BHPs enter the formulation because the overall source term q_j^w (which is a phase injection or production rate) is represented in terms of the standard well equation:

$$q_j^w = \sum_{s=1}^{n_s} WI_s \lambda_{j,s} (p_s - p_{w,s}), \tag{16}$$

where n_s is the number of blocks in which the well is completed (open to flow), WI_s is the well index for block s (which is essentially the transmissibility between the well and the block), $\lambda_{j,s}$ is the phase mobility in block s, p_s is the well-block pressure, and $p_{w,s}$ is the wellbore pressure in block s. BHP corresponds to the wellbore pressure at a particular location, such as the first well completion. If we have a horizontal well and pressure losses along the well are neglected (as they are here), then $p_{w,s}$ is constant along the well. Once the primary variables in well blocks are determined and a flash has been performed, well flow rates can be calculated using Eq. 16 (these flow rates are at subsurface conditions; to determine rates at surface conditions an additional flash must be performed).

For cases with well rate (rather than BHP) specifications, we require matrices that are analogous to $\mathbf{J}_{RR}$ and $\mathbf{J}_{RW}$ in Eq. 15. We use a tilde for cases where well rates are specified, and denote the extended Jacobian matrix $\tilde{\mathbf{J}}_{\text{ext}}$ as:

$$\tilde{\mathbf{J}}_{\text{ext}} = \begin{bmatrix} \tilde{\mathbf{J}}_{RR} & \tilde{\mathbf{J}}_{RC} \end{bmatrix}, \tag{17}$$

where, as in Eq. 15, $\tilde{\mathbf{J}}_{RR}$ ($2n_b \times 2n_b$) is the derivative of the reservoir equations with respect to primary reservoir variables, and $\tilde{\mathbf{J}}_{RC}$ ($2n_b \times n_u$, where n_u is the number of controls) is the derivative of the reservoir equations with respect to rate-control variables. Here we specify one rate control per well, so $n_u = n_w$.

The matrix $\tilde{\mathbf{J}}_{RC}$ is different from $\mathbf{J}_{RW}$ in Eq. 15 because the problem now involves derivatives of the reservoir equations with respect to rate controls instead of BHPs. Less obvious is the fact

that $\tilde{\mathbf{J}}_{RR}$ also differs from $\mathbf{J}_{RR}$. This is because, when we specify a well flow rate q_j^w, the well block pressure (p_s in Eq. 16), which is a primary reservoir variable, does not appear in the source term. By contrast, when BHP is specified, p_s does appear, so there is an additional contribution in $\mathbf{J}_{RR}$ that is not in $\tilde{\mathbf{J}}_{RR}$.

In concept, it should be straightforward to directly construct $\tilde{\mathbf{J}}_{RR}$ and $\tilde{\mathbf{J}}_{RC}$, just as we do for $\mathbf{J}_{RR}$ and $\mathbf{J}_{RW}$. This is nontrivial, however, because of the way in which AD-GPRS represents wells under rate control. Specifically, in this case AD-GPRS still represents q_j^w using Eq. 16, but it also includes additional equations (well equations) that relate q_j^w and BHP. These equations are a simple rearrangement of Eq. 16; i.e.,

$$\sum_{s=1}^{n_s} WI_s \lambda_{j,s}(p_s - p_{w,s}) - q_j^w = 0, \qquad j = g,\ w. \tag{18}$$

Because more equations are now treated in AD-GPRS, additional derivatives appear. We thus define the 'augmented' Jacobian matrix $\mathbf{J}^*$ as:

$$\mathbf{J}^* = \begin{bmatrix} \mathbf{J}_{RR} & \mathbf{J}_{RW} & \mathbf{J}_{RC} \\ \mathbf{J}_{WR} & \mathbf{J}_{WW} & \mathbf{J}_{WC} \end{bmatrix}, \tag{19}$$

where $\mathbf{J}_{RR}$ and $\mathbf{J}_{RW}$ are as defined in Eq. 15, $\mathbf{J}_{RC}$ ($2n_b \times n_u$) is the derivative of the reservoir equations with respect to the rate-control variables, $\mathbf{J}_{WR}$ ($n_w \times 2n_b$) is the derivative of the well equations with respect to reservoir variables, $\mathbf{J}_{WW}$ ($n_w \times n_w$) is the derivative of the well equations with respect to well BHPs, and $\mathbf{J}_{WC}$ ($n_w \times n_u$) is the derivative of the well equations with respect to rate-control variables. Because the rate-control variable q_j^w does not appear explicitly in the reservoir equation (due to the AD-GPRS treatment), we have $\mathbf{J}_{RC} = \mathbf{0}$. In addition, because the derivative of the well equation (Eq. 18) with respect to its rate specification q_j^w is -1, the matrix $\mathbf{J}_{WC}$ in Eq. 19 is simply the negative identity matrix (i.e., $\mathbf{J}_{WC} = -\mathbf{I}$).

In order to determine the desired matrices $\tilde{\mathbf{J}}_{RR}$ and $\tilde{\mathbf{J}}_{RC}$, we apply a Schur complement procedure to $\mathbf{J}^*$ in Eq. 19. This entails premultiplying $\mathbf{J}^*$ by a matrix $\mathbf{S}$, with the goal of eliminating $\mathbf{J}_{RW}$. Defining

$$\mathbf{S} = \begin{bmatrix} \mathbf{I} & -\mathbf{J}_{RW}\mathbf{J}_{WW}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{20}$$

and constructing a modified 'augmented' Jacobian matrix $\tilde{\mathbf{J}}^* = \mathbf{S}\mathbf{J}^*$, we have:

$$\tilde{\mathbf{J}}^* = \begin{bmatrix} \mathbf{J}_{RR} - \mathbf{J}_{RW}\mathbf{J}_{WW}^{-1}\mathbf{J}_{WR} & \mathbf{0} & \mathbf{J}_{RC} - \mathbf{J}_{RW}\mathbf{J}_{WW}^{-1}\mathbf{J}_{WC} \\ \mathbf{J}_{WR} & \mathbf{J}_{WW} & \mathbf{J}_{WC} \end{bmatrix}. \tag{21}$$

We can now partition $\tilde{\mathbf{J}}^*$ into submatrices as:

$$\tilde{\mathbf{J}}^* = \begin{bmatrix} \tilde{\mathbf{J}}_{RR} & \tilde{\mathbf{J}}_{RW} & \tilde{\mathbf{J}}_{RC} \\ \tilde{\mathbf{J}}_{WR} & \tilde{\mathbf{J}}_{WW} & \tilde{\mathbf{J}}_{WC} \end{bmatrix}. \tag{22}$$

Comparing this with Eq. 21, we see that

$$\tilde{\mathbf{J}}_{RR} = \mathbf{J}_{RR} - \mathbf{J}_{RW}\mathbf{J}_{WW}^{-1}\mathbf{J}_{WR}, \tag{23}$$

$$\tilde{\mathbf{J}}_{RW} = \mathbf{0}, \tag{24}$$

$$\tilde{\mathbf{J}}_{RC} = \mathbf{J}_{RC} - \mathbf{J}_{RW}\mathbf{J}_{WW}^{-1}\mathbf{J}_{WC}, \tag{25}$$

where $\mathbf{J}_{RC} = \mathbf{0}$ and $\mathbf{J}_{WC} = -\mathbf{I}$, as noted earlier. Eq. 24 indicates that the direct dependency of the reservoir equations on well BHPs has been eliminated (which was our goal since wells are now controlled by rates). Importantly, the matrices $\tilde{\mathbf{J}}_{RR}$ and $\tilde{\mathbf{J}}_{RC}$ are the derivatives required for $\tilde{\mathbf{J}}_{\text{ext}}$ in Eq. 17 for well-rate control problems.

In our actual implementation, AD-GPRS provides $\mathbf{J}^*$ in Eq. 19. The required matrices $\tilde{\mathbf{J}}_{RR}$ and $\tilde{\mathbf{J}}_{RC}$ are computed by performing the calculations shown in Eqs. 23 and 25. These computations are inexpensive since n_w is small.

3 Numerical Simulation Results

In this section, we present results using the POD-TPWL model for CO_2-water systems, with both BHPs and rates as control parameters. Two examples will be considered — one is a large-scale synthetic aquifer model, and the other is a simplified model of the CO_2 injection site that was planned for FutureGen 2.0. As noted in Section 2, compared with previous applications of POD-TPWL [39, 41], new features included here are the use of horizontal wells, the application of well rate specifications, and the use of a grid with blocks that vary significantly in size over the domain.

We note that POD-TPWL results for a CO_2 EOR (compositional) problem, which includes the use of horizontal wells, are presented in Appendix B.

3.1 Model 1: Synthetic Aquifer

The problem set up for this case is based on a previous model used by Cameron & Durlofsky [13] to optimize carbon storage operations. However, many of the specifics, including the permeability field and the well settings, differ from those used in [13].

3.1.1 Problem Set Up

The simulation model, shown in Fig. 1, includes a storage aquifer, of physical dimensions 10.9 km $\times$ 10.9 km $\times$ 100 m, immersed within a large-scale regional model, of physical dimensions 232 km $\times$ 232 km $\times$ 100 m. The storage aquifer is modeled on a $25 \times 25 \times 10$ grid (total of 6250 grid blocks),

while the full system is represented on a $39 \times 39 \times 10$ grid (total of 15,210 grid blocks). In the storage aquifer, grid blocks are of size 436 m $\times$ 436 m $\times$ 10 m. Block sizes increase as we move out from the storage aquifer, as is evident in Fig. 1 (left). Note that the grid is still fairly dense just outside the storage aquifer. CO_2 injection is accomplished using two horizontal wells, as shown in Fig. 1 (right).

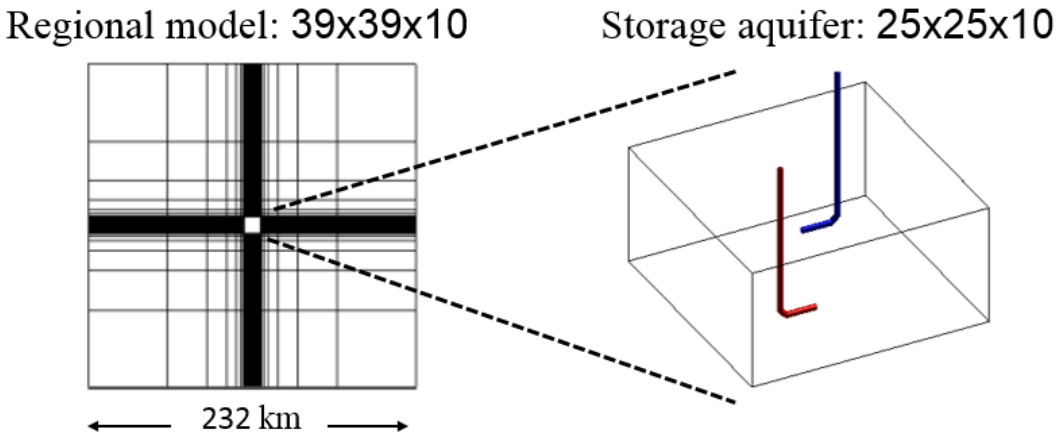
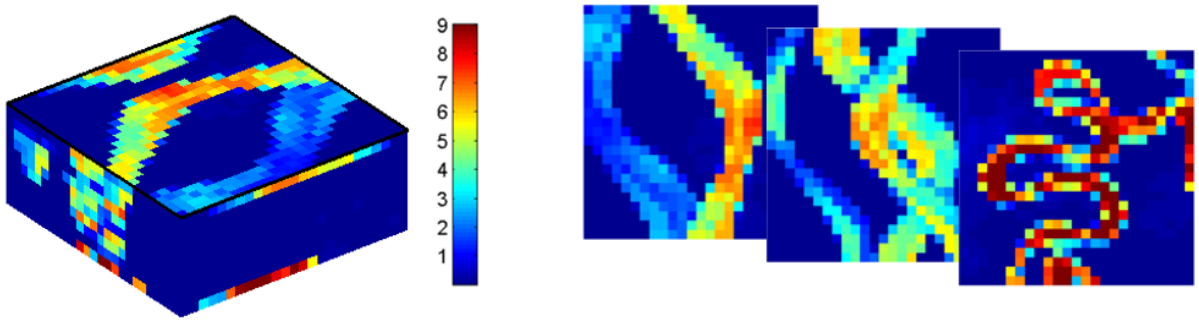


Figure 1: Simulation grid (left, areal view) and horizontal injection wells (right)

The permeability and porosity fields are sampled from the Stanford VI synthetic geological model [20], which represents a highly heterogeneous channelized system. Fig. 2 displays $\log k$ for this system (permeability is isotropic so $\mathbf{k} = k\mathbf{I}$). The vertical variation in the channel structure is evident in Fig. 2b.



(a) Horizontal permeability (b) Layered view

Figure 2: Permeability field for storage aquifer ($\log k$ is shown)

As discussed in Section 2, we use a compositional simulation model with two components (CO_2 and water) and two phases (gas and water). Thus, this system contains $15,210 \times 2 = 30,420$ primary variables. The relative permeabilities for the water and gas phases are defined using the

Brooks-Corey relation. We set residual gas saturation $S_{gr} = 0.1$, irreducible water saturation $S_{wi} = 0$, and use exponents of 2 for both phases. Capillary pressure is neglected. The (isothermal) aquifer temperature is set to 372 K; surface conditions correspond to temperature and pressure of 293 K and 1.013 bar.

The initial aquifer pressure at the top layer is 170 bar. The overall molar fractions of the initial in-situ fluid are 0.001 CO_2 and 0.999 water. The injected fluid is 0.999 CO_2 and 0.001 water. The two horizontal CO_2 injection wells are located in layers 7 and 8 of the storage aquifer (near the bottom of the model). Each of the wells is completed in three grid blocks. The model is run for a total of 10,000 days, which is about 27 years. We will present results using both BHP and rate specifications. The cumulative injected volume of CO_2 is about 1% - 4% of the pore volume of the storage aquifer in cases with rate control. Much larger volumes are injected in cases with BHP control, which is not consistent with practical operations but is useful for purposes of testing POD-TPWL for this problem.

3.1.2 POD-TPWL Results with BHP Controls (Model 1)

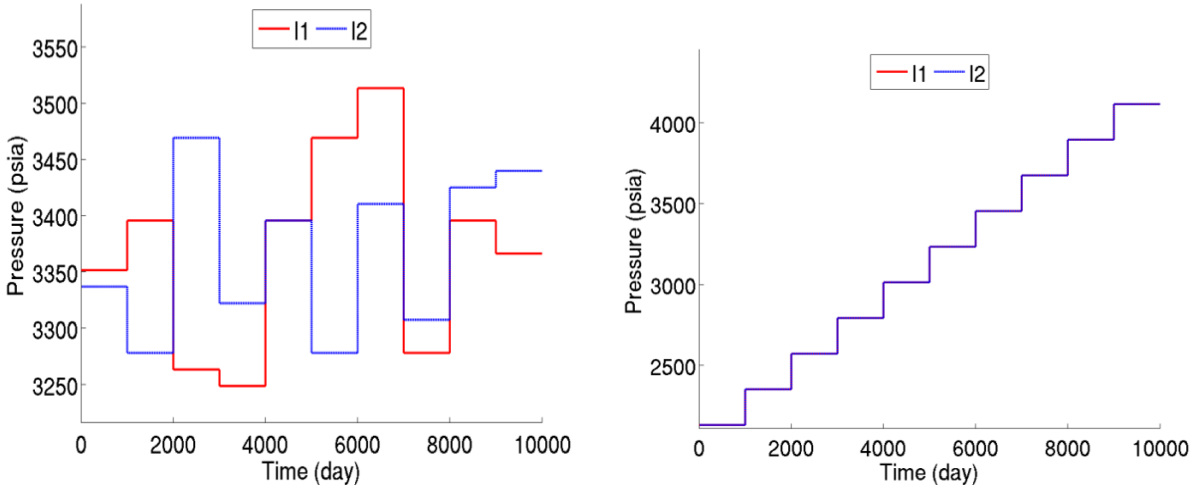
In our application of POD-TPWL, we first perform two full-order training runs, which are used to construct the POD-TPWL model (one of these is the primary training run, as discussed in Section 2.2). We then perform a sequence of test runs, where the time-varying BHP controls differ from those used in the training runs. As in [41], the degree of perturbation from the primary training run is quantified using the parameter α. A value of $\alpha = 0$ indicates the training case (which the POD-TPWL model should match exactly), and $\alpha = 1$ indicates the case with the largest perturbation, referred to as the target case. For values of α between 0 and 1, the test-case BHP for a particular well at time t (designated $\mathbf{u}^t_{\text{test}}$) is given by:

$$\mathbf{u}^t_{\text{test}} = (1-\alpha)\mathbf{u}^t_{\text{training}} + \alpha \mathbf{u}^t_{\text{target}}, \tag{26}$$

where $\mathbf{u}^t_{\text{training}}$ and $\mathbf{u}^t_{\text{target}}$ are the training and target BHPs for the well at time t. Figs. 3a and b show the time-varying BHPs for the training and target runs. The BHPs in the training case are generated randomly, while the BHPs in the target case increase stepwise-linearly in time. These BHP profiles are intended to represent the types of time-varying BHP schedules that might be computed from an optimization procedure (such as that reported in [13]).

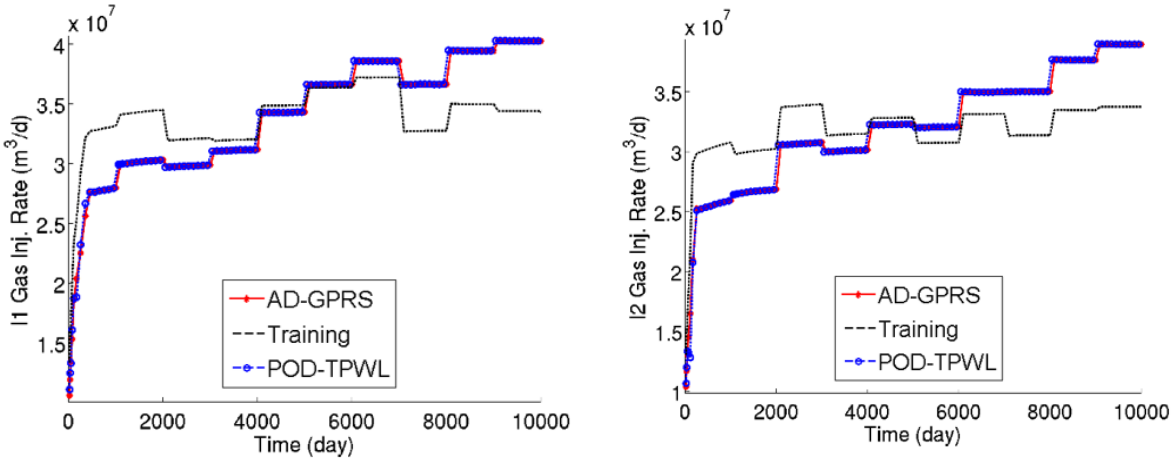
A total of 264 snapshots are collected from the two full-order training runs. In the POD-TPWL model, we use $l_p = 90$ reduced pressure variables and $l_z = 120$ reduced overall water molar fraction variables, for a total of $l = 210$ variables. This represents a substantial reduction compared to the reference AD-GPRS model, which entails 30,420 primary variables.

Figs. 4, 5 and 6 display the CO_2 injection rates (at surface conditions) for the two horizontal injectors for three different test cases, which correspond to $\alpha = 0.3, 0.5$ and 0.8. In these and subsequent plots, the black dotted curves represent results for the primary training case (these are the results around which we linearize to compute the solution for the POD-TPWL test case), the



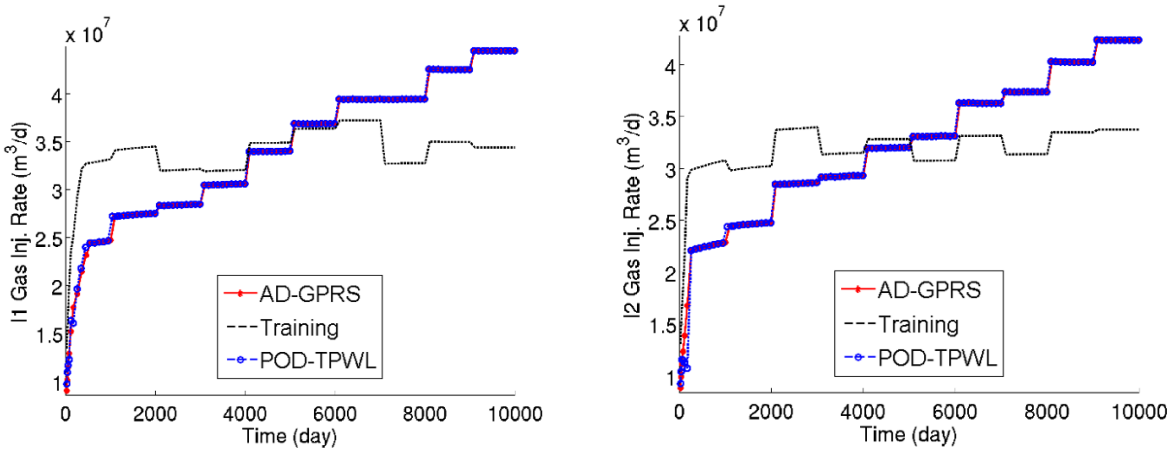
(a) Training run ($\alpha = 0$) (b) Target run ($\alpha = 1$)

Figure 3: Time-varying BHPs for training and target simulations (Model 1)



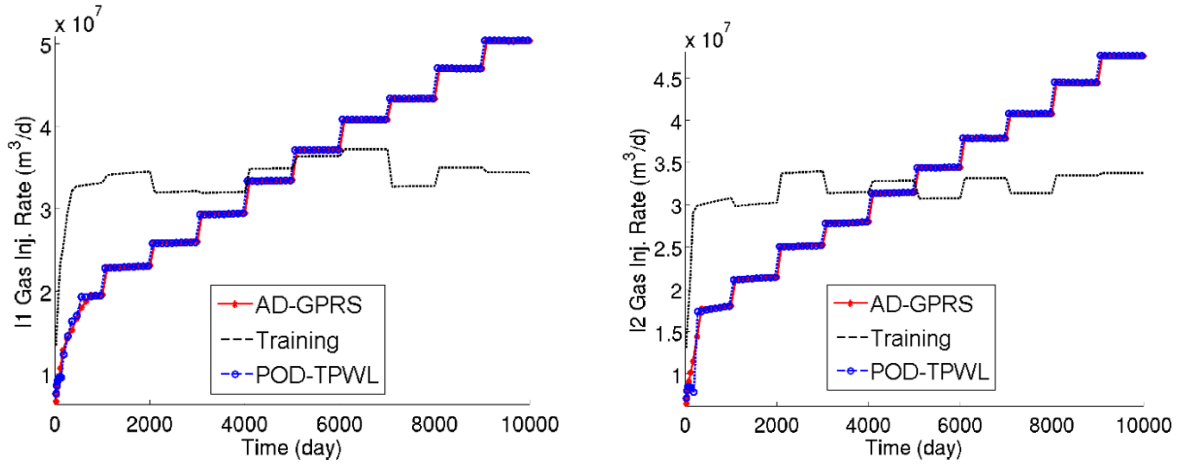
(a) Well 1 (b) Well 2

Figure 4: CO_2 injection rates for test case with $\alpha = 0.3$ (Model 1)



(a) Well 1 (b) Well 2

Figure 5: CO_2 injection rates for test case with $\alpha = 0.5$ (Model 1)



(a) Well 1 (b) Well 2

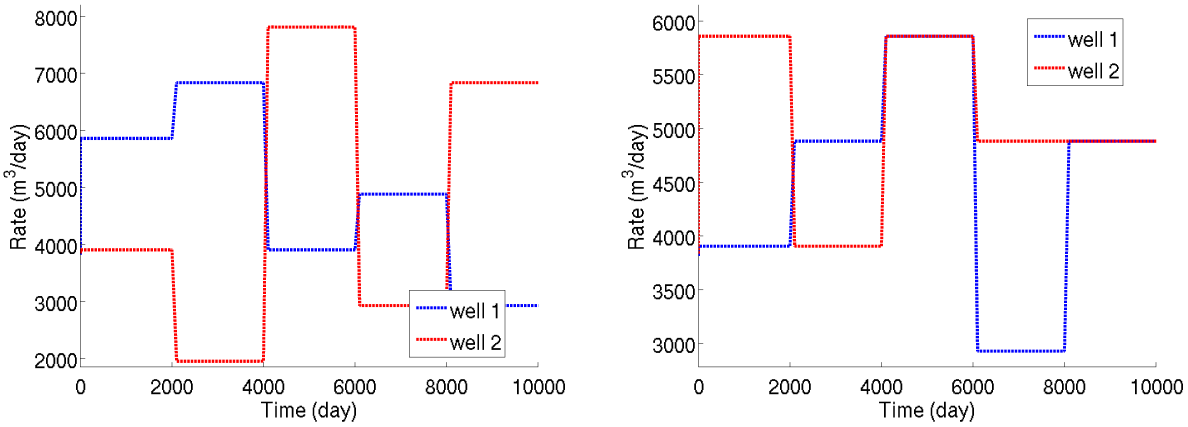
Figure 6: CO_2 injection rates for test case with $\alpha = 0.8$ (Model 1)

red curves define the full-order reference AD-GPRS solution for the test case, and the blue curves depict the POD-TPWL results.

From these figures, we see that there are, in general, considerable differences between the training and test solutions. These differences increase with increasing α, as would be expected. Note also that the injection rates at later times in the test runs are significantly higher than those in the training run. The POD-TPWL results are, however, in consistently close agreement with the full-order solution in all three cases. This is encouraging, as it suggests that POD-TPWL may indeed be suitable for use in CO_2 storage applications.

3.1.3 POD-TPWL Results with Rate Controls (Model 1)

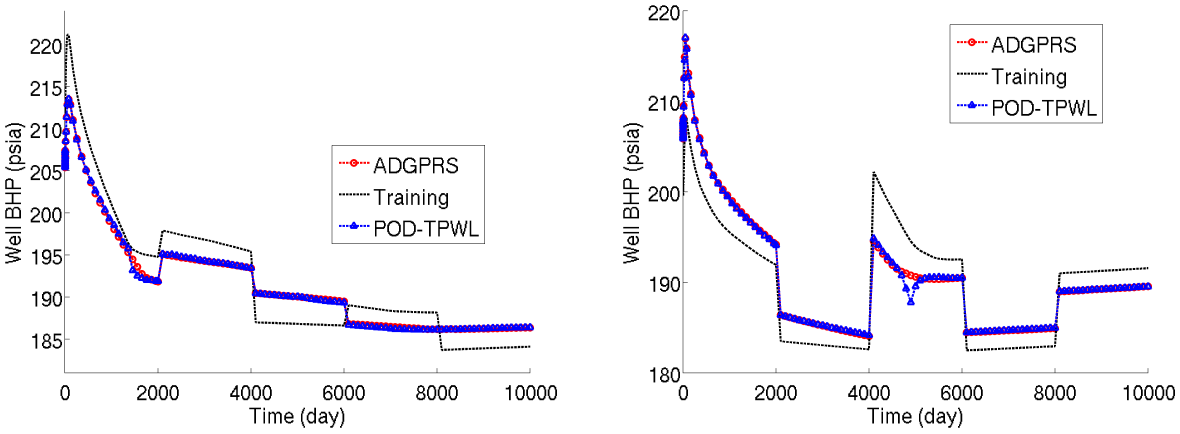
We now consider cases with rate specifications. We again test the POD-TPWL model using linear combinations of training ($\alpha = 0$) and target ($\alpha = 1$) runs. The rate profiles at subsurface conditions (372 K, $\sim$ 13.5 bar) are shown in Fig. 7. The perturbations applied in this case are larger than those applied to the BHP controls in Section 3.1.2. The total injection rate for each well is about 1.5 million metric tons of CO_2 per year. The cumulative CO_2 injected after 10,000 days is close to 3% of the pore volume of the storage aquifer. The POD basis matrix is constructed from 254 snapshots collected from two full-order training runs. For this case we use $l_p = l_z = 80$.



(a) Training run ($\alpha = 0$) (b) Target run ($\alpha = 1$)

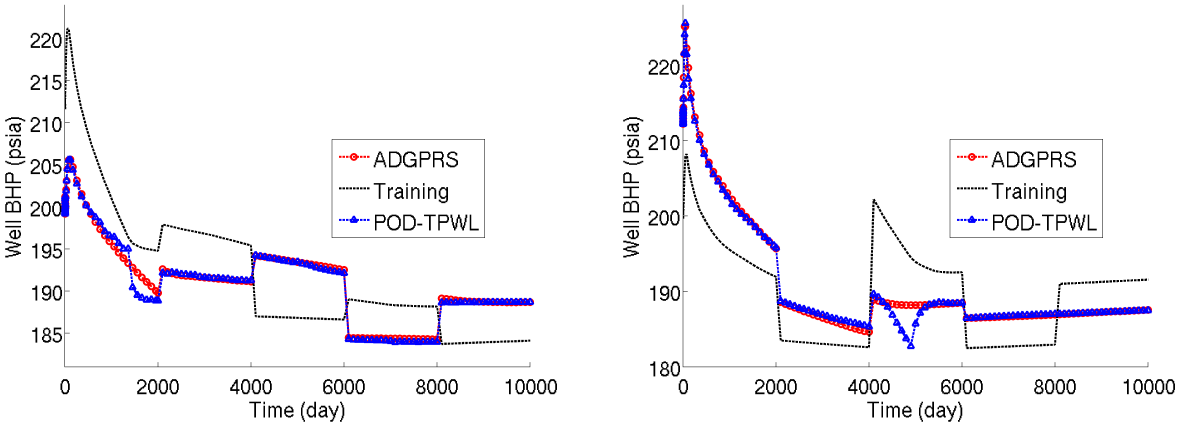
Figure 7: Time-varying rate specifications for training and target simulations (Model 1)

Since we now prescribe injection rates, the relevant well quantity to assess is BHP. Injection well BHPs for test cases with $\alpha = 0.5$ and 1.0 are shown in Figs. 8 and 9. In general, the POD-TPWL results (blue curves) display reasonable accuracy relative to the reference full-order simulation (red curves). As α increases, POD-TPWL error is seen to increase. In Fig. 9b, inaccuracies are observed in the POD-TPWL model for Well 2 at around 5000 days (this is also evident in Fig. 8b). We believe that errors of this type occur due to our point selection treatment. Basically, we do not allow the time step corresponding to the selected training-run point to decrease in time. This is generally an appropriate restriction, but it can lead to inaccuracy at some time steps.



(a) Well 1 (b) Well 2

Figure 8: CO_2 injection well BHPs for test case with $\alpha = 0.5$ (Model 1)

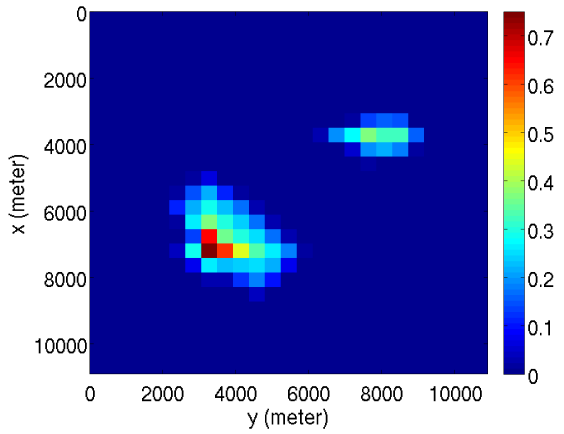


(a) Well 1 (b) Well 2

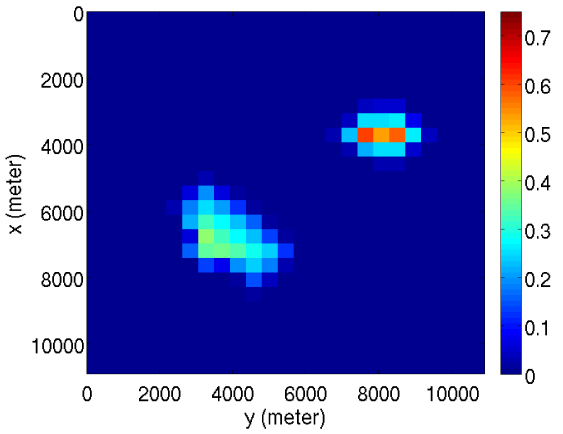
Figure 9: CO_2 injection well BHPs for test case with $\alpha = 1.0$ (Model 1)

For the $\alpha = 1.0$ test case, we also present, in Fig. 10, color maps for overall molar fraction of CO_2 at 4000 days. These results are shown for layers 7 and 8, which contain the two injection wells. Red indicates high CO_2 concentration, and blue denotes high water concentration. The CO_2 distribution for the training run is shown in Figs. 10a and b, AD-GPRS (full-order) test-case results are shown in Figs. 10c and d, and POD-TPWL results are presented in Figs. 10e and f. The differences between the training and test runs are relatively small, but it is evident that POD-TPWL is able to provide results that (fairly closely) resemble the full-order test runs.

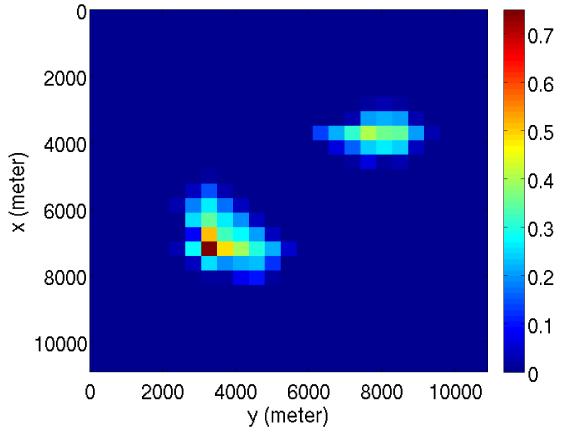
Finally, in Fig. 11, we compare the states (z_g) generated from POD-TPWL to those from AD-GPRS. These results are at times of 4000 days and 10,000 days. The points fall reasonably close to the 45-degree line (this line indicates perfect agreement), though there is some scatter, especially for small z_g. Overall, however, this plot indicates that POD-TPWL is able to provide reasonable



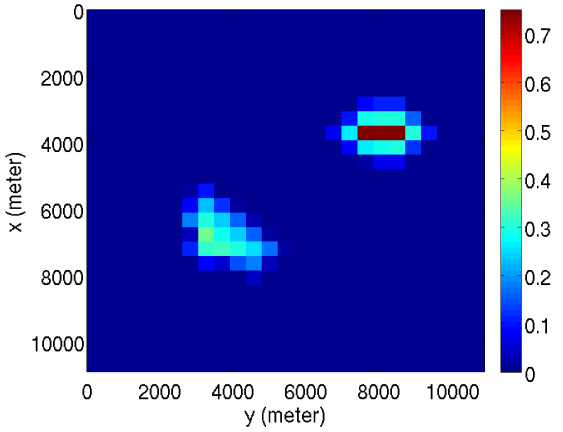
(a) Training run, layer 7



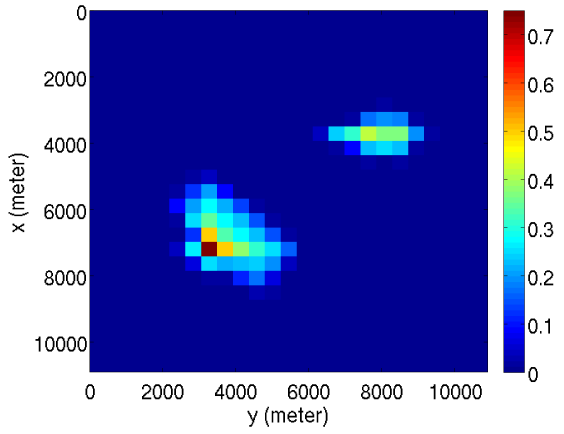
(b) Training run, layer 8



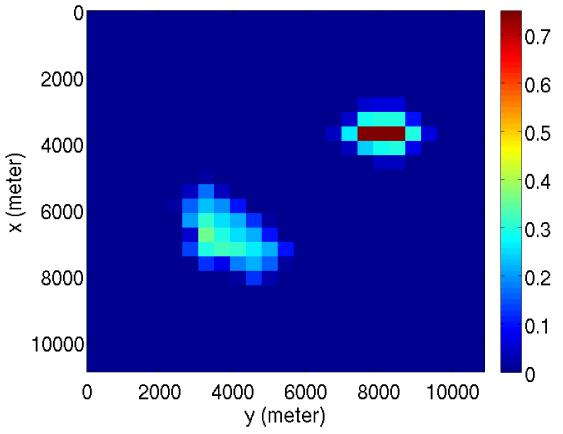
(c) Test case with AD-GPRS, layer 7



(d) Test case with AD-GPRS, layer 8



(e) Test case with POD-TPWL, layer 7

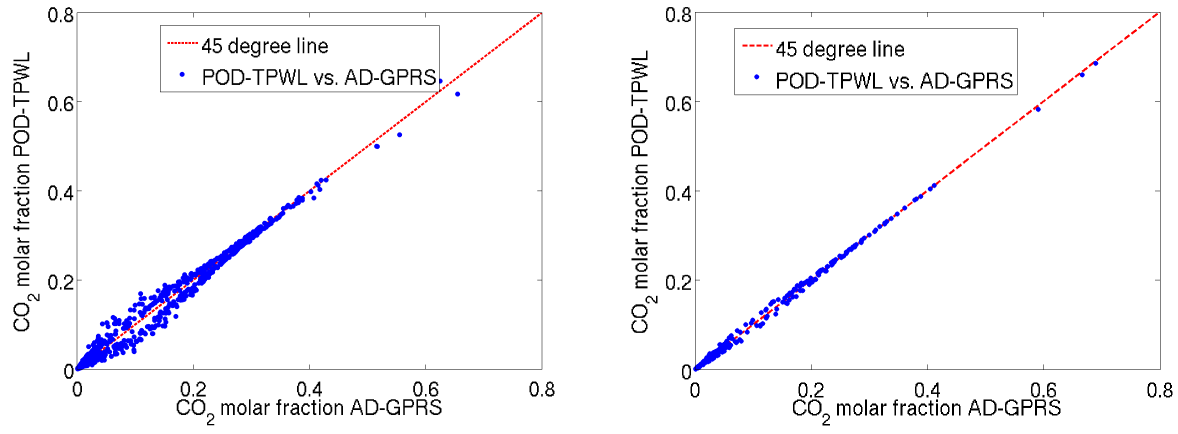


(f) Test case with POD-TPWL, layer 8

Figure 10: Color maps for CO_2 overall molar fraction at 4000 days with $\alpha = 1.0$ (Model 1)

estimates of the z_g state variable.

For this case, the full-order AD-GPRS simulations typically require around 18 minutes to run on one compute node with 16 cores with Intel® dual Sandy Bridge™ CPUs. POD-TPWL models require only about 3 seconds on one core, which represents a runtime speedup factor of about 360. We reiterate that POD-TPWL model construction (the offline portion of the procedure) entails computation corresponding to about 2.3 full-order runs.



(a) 4000 days (b) 10,000 days

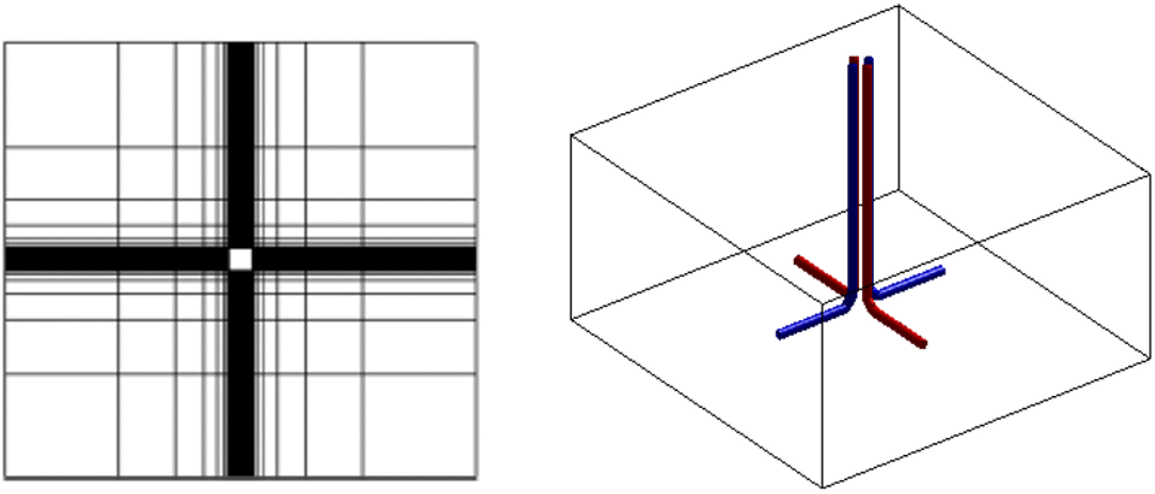
Figure 11: Comparison of CO_2 overall molar fraction between POD-TPWL and AD-GPRS at 4000 days and 10,000 days for test case with $\alpha = 1.0$ (Model 1)

3.2 Model 2: Mount Simon Formation

The site proposed for the CO_2 storage associated with DOE's FutureGen 2.0 project is located in Morgan County, Illinois [7]. The target CO_2 injection zone was a high-permeability region within the upper portion of the Mount Simon sandstone. Unfortunately, after over four years of development, this project has been canceled. However, the substantial amount of data collected and the modeling performed to date are still valuable. The model considered in this section is a simplified version of the Mount Simon simulation model constructed by Bonneville et al. [7].

3.2.1 Problem Set Up

The simulation model is shown in Fig. 12. It includes a storage aquifer of dimensions 3.1 mi (5 km) $\times$ 3.1 mi (5 km) $\times$ 1346 ft (410 m), which is immersed within a large-scale regional model of dimensions 100 mi $\times$ 100 mi $\times$ 1346 ft. The storage aquifer is modeled on a $30 \times 30 \times 30$ grid (total of 27,000 grid blocks), while the full system is represented on a $46 \times 46 \times 30$ grid (total of 63,480 grid blocks). In the storage aquifer, grid blocks are 525 ft (160 m) on a side. Grid-block thickness varies significantly, from 10 ft to 190 ft. CO_2 injection is accomplished using four horizontal wells (consistent with proposed operations [7]), as shown in Fig. 12b. The permeability and porosity fields are completely layered, so there is property variation only in the vertical direction (the model is shown in Fig. 13). For this case, we take $k_x = k_y$ and $k_z = 0.1k_x$. This model represents a less detailed version of the Mount Simon geological model considered in [7], which contains 51 layers.



(a) Storage aquifer and regional model (b) Four horizontal wells

Figure 12: Areal grid and well locations for simplified Mount Simon model

Our simulation model again contains CO_2 and water components in gas and water phases. The initial aquifer pressure is 124 bar. The four horizontal wells are located in layer 25 of the model. Injection wells I1 and I4 are of length 1575 ft (480 m), and wells I2 and I3 are of length 2625 ft (800 m). We control the wells by specifying both time-varying BHPs and rates. The model is run

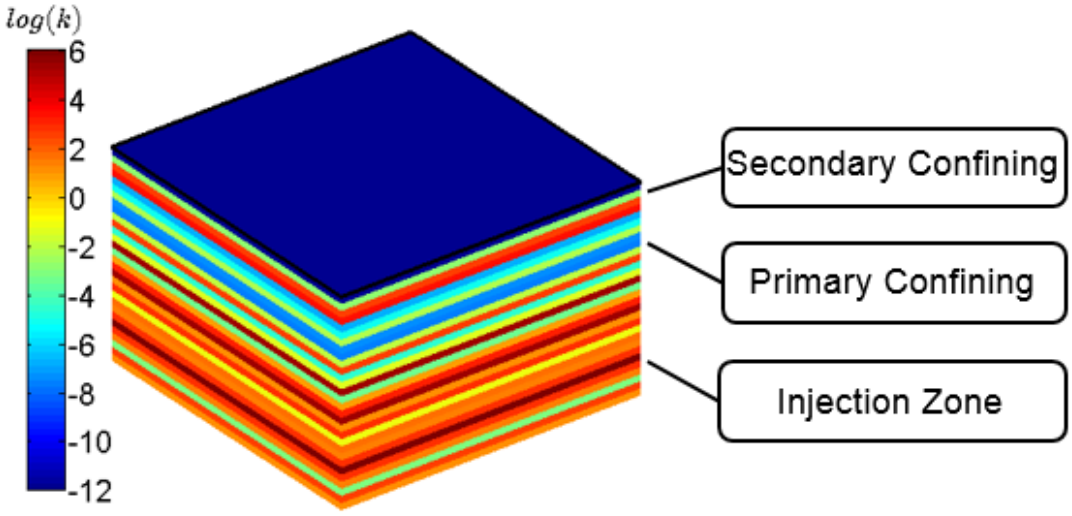
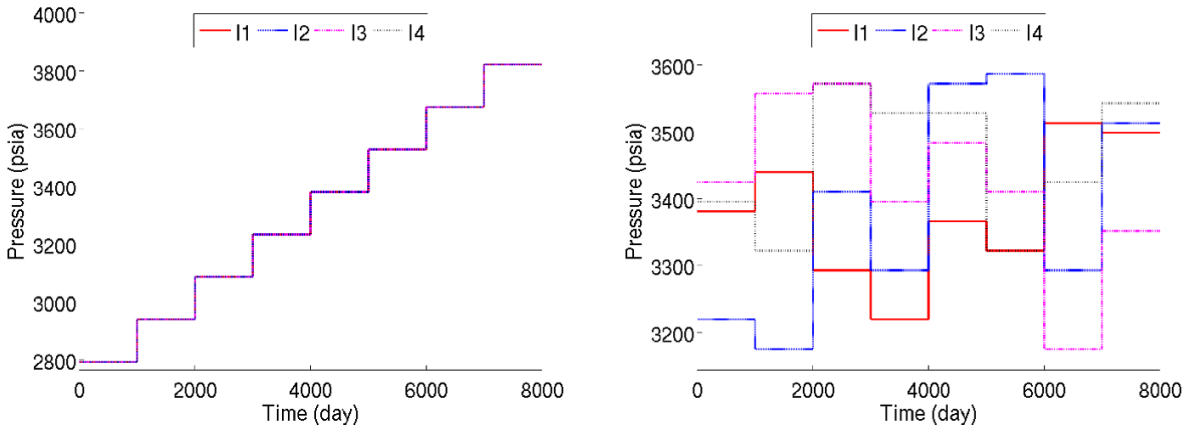


Figure 13: Permeability field for simplified Mount Simon model ($\log k_x$ is shown, k_x in mD)

for a total of 8000 days (about 22 years) in BHP-control cases, and for 7300 days (about 20 years) in rate-control cases. Other model properties are the same as in Model 1. The full-order system in this case contains $63,480 \times 2 = 126,960$ primary variables.

3.2.2 POD-TPWL Results with BHP Controls (Model 2)

This case involves training with BHPs that increase in time and testing with BHPs that include a random component. Figs. 14a and b show the time-varying BHPs for the training ($\alpha = 0$) and target ($\alpha = 1$) runs. Note that the target BHPs differ for each of the four injection wells. A total of 214 snapshots are collected from the two full-order training runs. In the POD-TPWL model, we set $l_p = 90$ and $l_z = 120$.

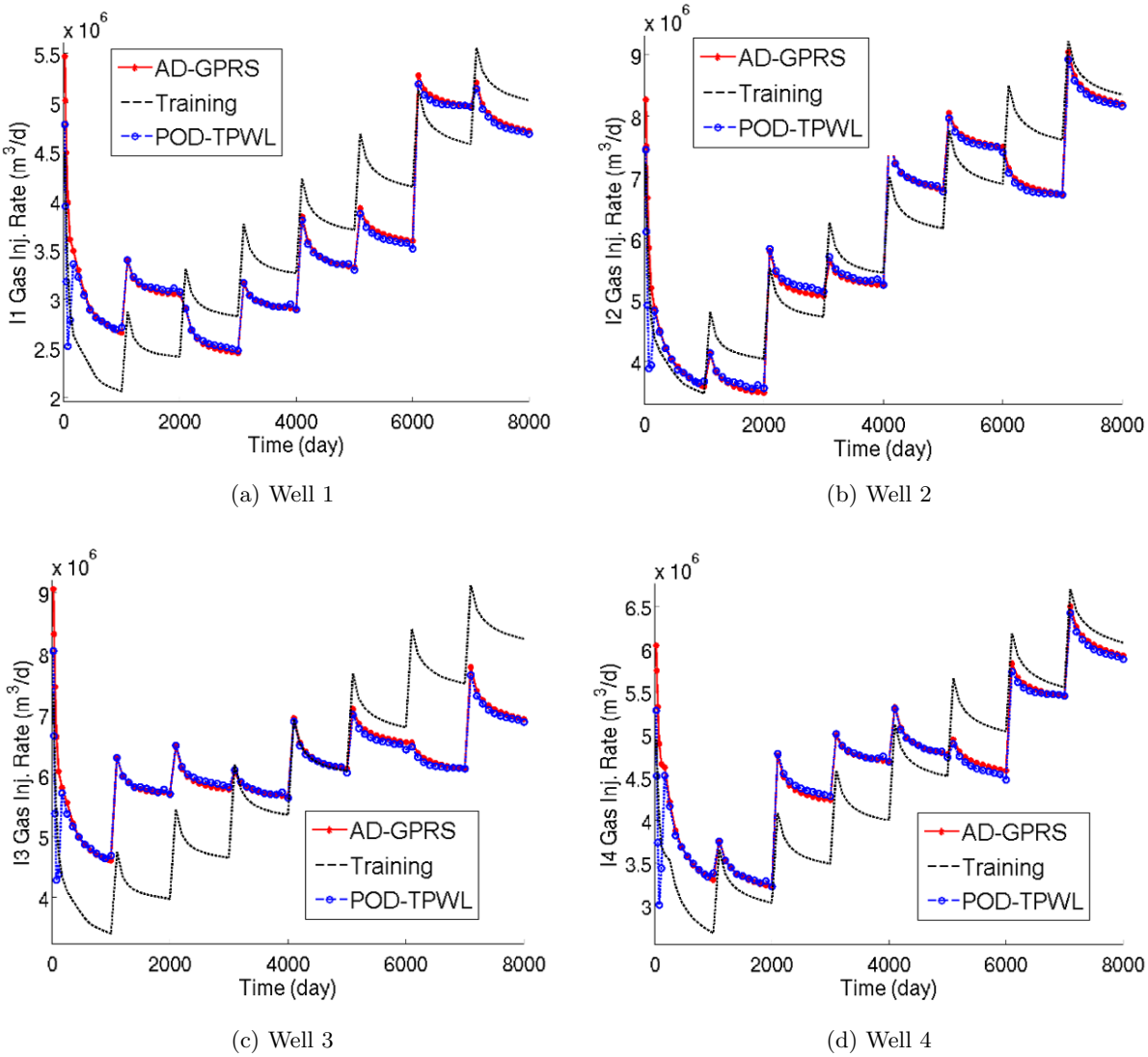


(a) Training case ($\alpha = 0$)

(b) Target case ($\alpha = 1$)

Figure 14: Time-varying BHPs for training and target simulations (Model 2)

Test-case results for $\alpha = 0.3$ and 0.8 are shown in Figs. 15 and 16. Time-varying injection rates are shown for all four wells. The POD-TPWL results in Fig. 15 are quite accurate relative to reference AD-GPRS results, though some error is apparent in Fig. 16. The general level of agreement is nonetheless relatively close, even though results for the training simulation differ considerably from those for the test run with $\alpha = 0.8$.



(a) Well 1

(b) Well 2

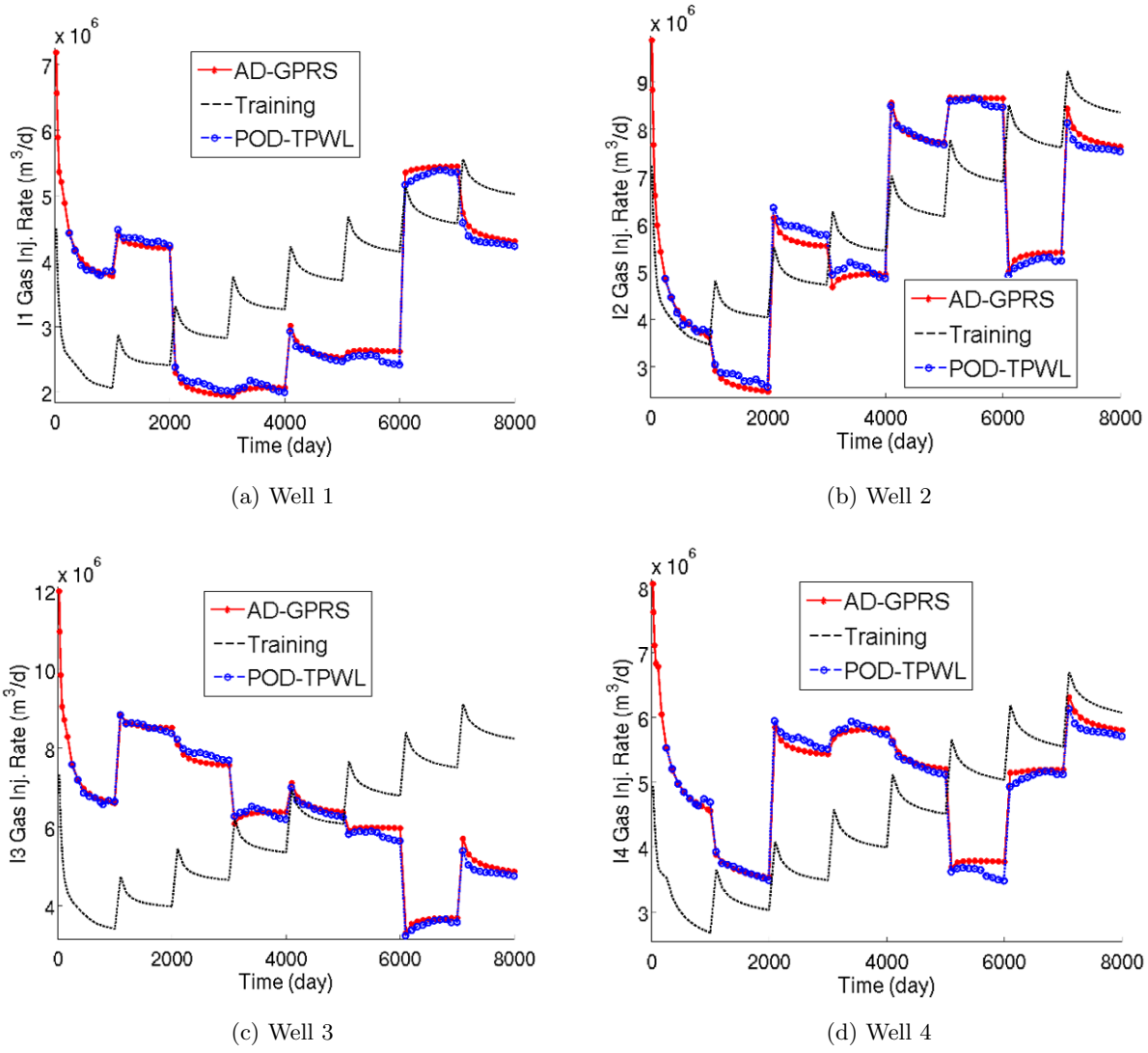
(c) Well 3

(d) Well 4

Figure 15: CO_2 injection rates for test case with $\alpha = 0.3$ (Model 2)

3.2.3 POD-TPWL Results with Rate Controls (Model 2)

We now consider results in which well injection rates are specified. The rate profiles at subsurface conditions (372 K, $\sim$ 14 bar) for both cases are shown in Fig. 17. In both the training and test runs, all wells are specified to inject the same volume of CO_2, though this is not a requirement of the implementation. The total injection rate for all four wells is approximately 1.1 million metric



(a) Well 1

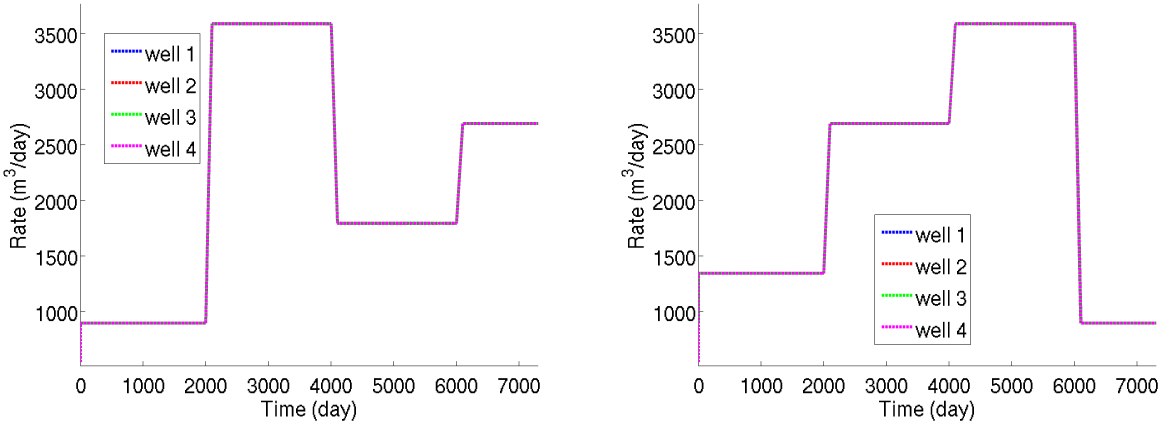
(b) Well 2

(c) Well 3

(d) Well 4

Figure 16: CO_2 injection rates for test case with $\alpha = 0.8$ (Model 2)

tons of CO_2 per year. The cumulative CO_2 injected after 7300 days is close to 3.5% of the pore volume of the storage aquifer. Here, we use $l_p = l_z = 80$ in the POD-TPWL model (a total of 200 snapshots were generated in the two full-order training runs).



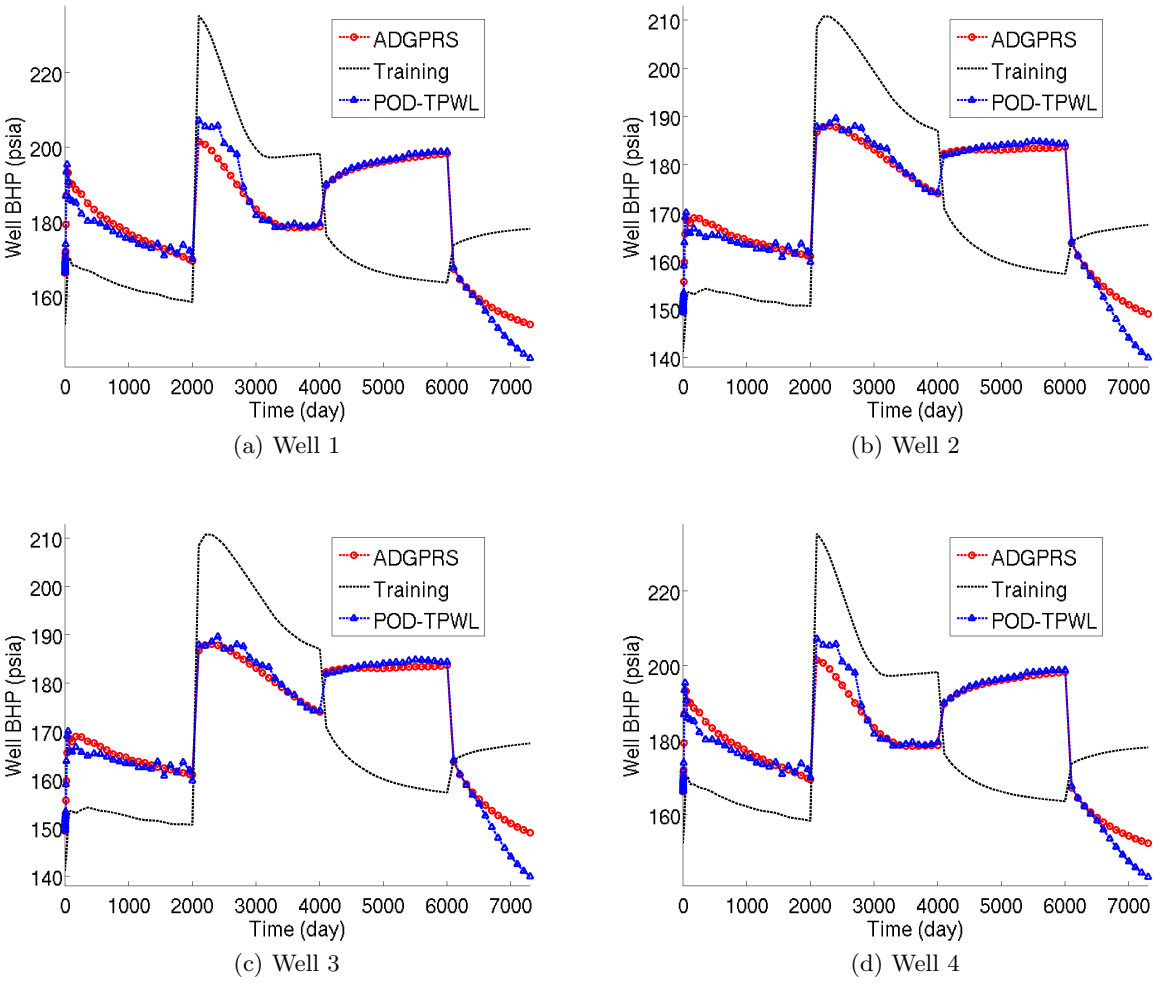
(a) Training case ($\alpha = 0$) (b) Target case ($\alpha = 1$)

Figure 17: Time-varying rate specifications for training and target simulations (Model 2)

For this case we present results only for $\alpha = 1$. Test-case results for injection well BHPs for the four wells are shown in Fig. 18. Overall, the POD-TPWL results display reasonable accuracy relative to AD-GPRS predictions, though some inaccuracy is observed at late time (after around 6500 days). We believe this is again due to the point selection scheme applied here. The development of a more robust point selection scheme will be considered in future work.

Color maps for CO_2 overall molar fraction (z_g) for the test case at 4000 days and 6000 days (for layer 25) are presented in Fig. 19. It is evident that z_g is high near the four injectors and that it decreases sharply as we move away from the wells. The differences between the full-order training and test runs are not that significant in these plots. We do, however, see that the POD-TPWL solution resembles the AD-GPRS solution fairly closely at 6000 days (Figs. 19d and f), where differences between the training and test runs are apparent.

The differences between the POD-TPWL and AD-GPRS solutions can be seen more directly by constructing difference maps. In Figs. 20a and b, we present the absolute value of the (block-by-block) difference in z_g between the AD-GPRS training and test simulations. Although the z_g scale spans a much smaller range here compared to that in Fig. 19, we see that there are clear differences between the training and test runs. Shown in Figs. 20c and d are differences between the AD-GPRS and POD-TPWL test solutions. If the POD-TPWL model was perfectly accurate, these plots would be dark blue. Error in z_g is clearly apparent, however, both in the near-well region where z_g is large, and at the edges of the plume, though it is relatively small in magnitude over most of the domain. The latter error may reflect the difficulty of capturing front locations precisely using POD-TPWL, which was also observed in [39]. It is possible, however, that fronts could be better resolved through use of a point selection scheme that gives more weight to front



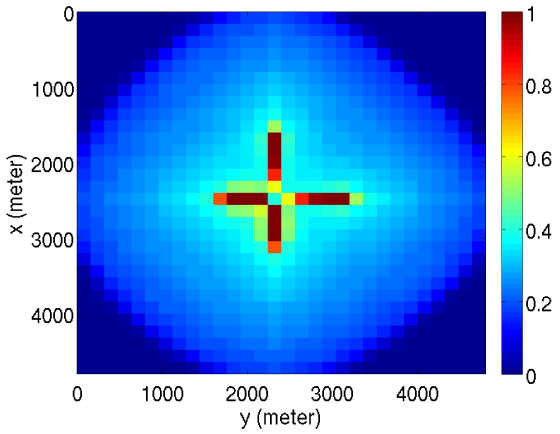
(a) Well 1

(b) Well 2

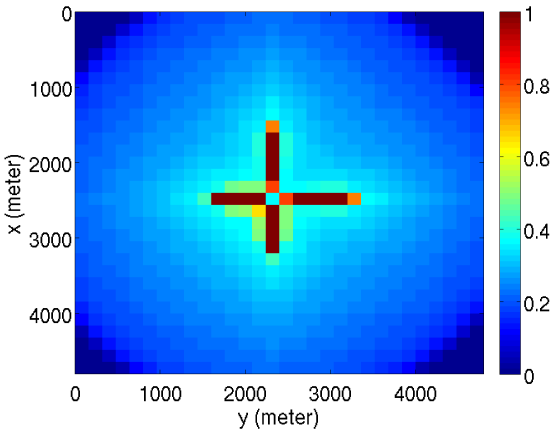
(c) Well 3

(d) Well 4

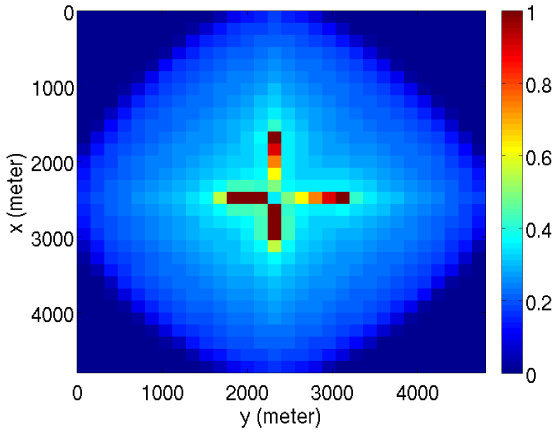
Figure 18: CO_2 injection well BHPs for test case with $\alpha = 1.0$ (Model 2)



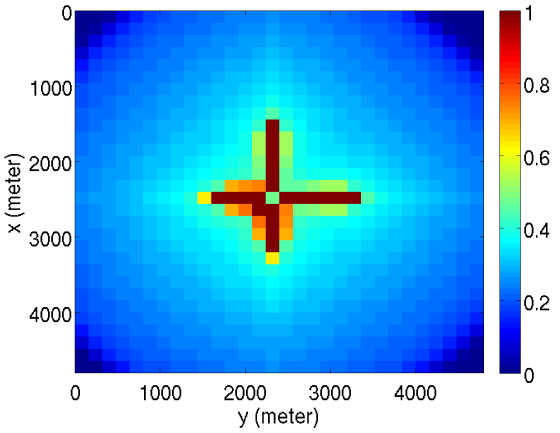
(a) Training run, 4000 days



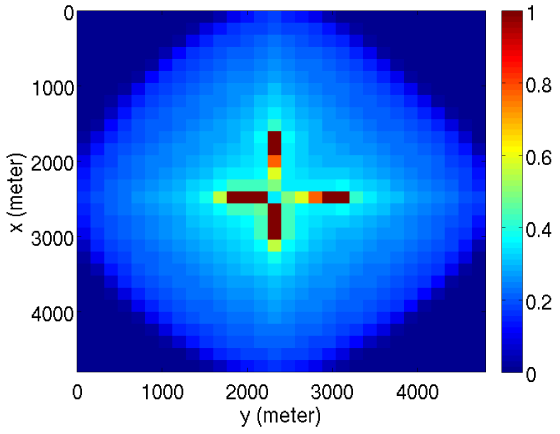
(b) Training run, 6000 days



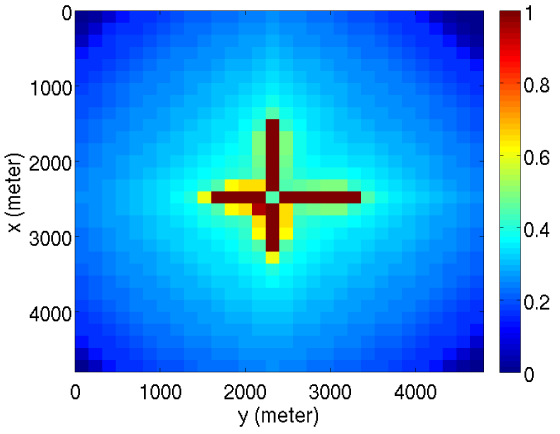
(c) Test with AD-GPRS, 4000 days



(d) Test with AD-GPRS, 6000 days

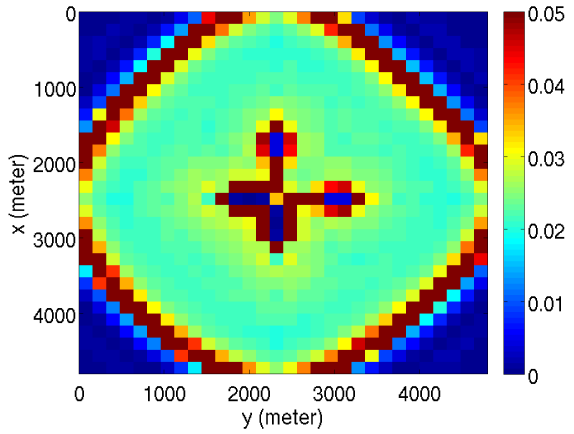


(e) Test with POD-TPWL, 4000 days

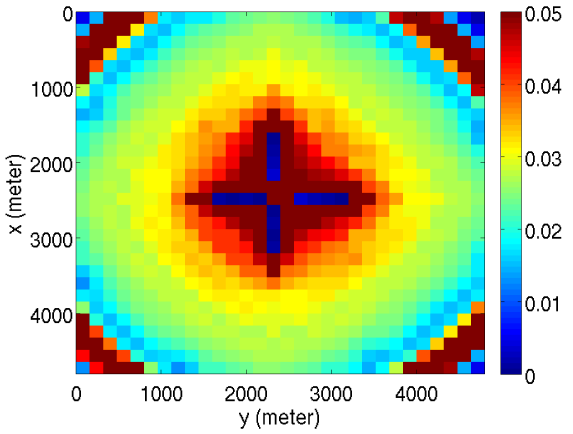


(f) Test with POD-TPWL, 6000 days

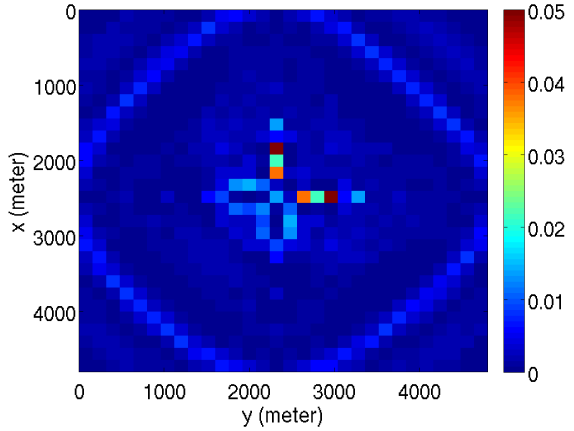
Figure 19: Color maps for CO_2 overall molar fraction z_g in layer 25 (Model 2)



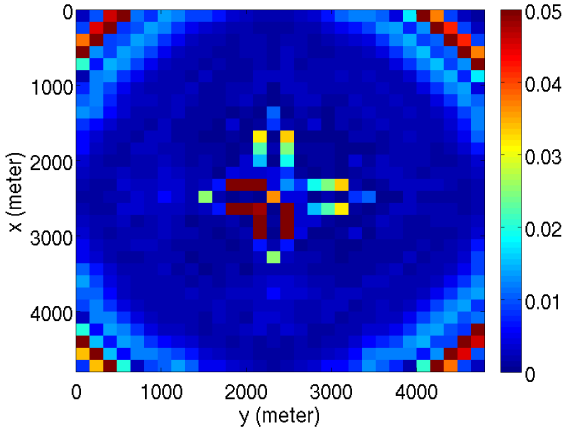
(a) Difference between AD-GPRS training and test runs, 4000 days



(b) Difference between AD-GPRS training and test runs, 6000 days



(c) Difference between POD-TPWL and AD-GPRS test runs, 4000 days



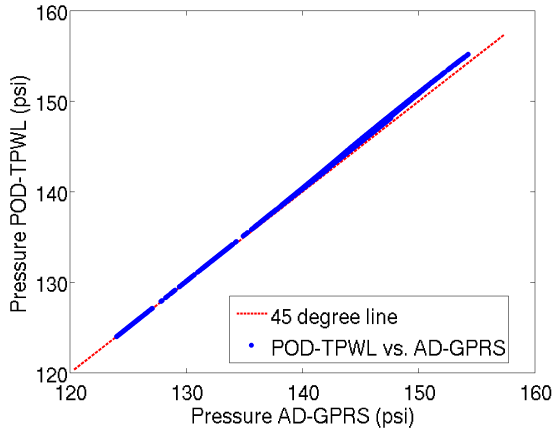
(d) Difference between POD-TPWL and AD-GPRS test runs, 6000 days

Figure 20: Absolute differences in z_g in layer 25 at 4000 and 6000 days (Model 2). Upper plots display differences between full-order training and test results, while lower plots display differences between POD-TPWL and AD-GPRS test solutions

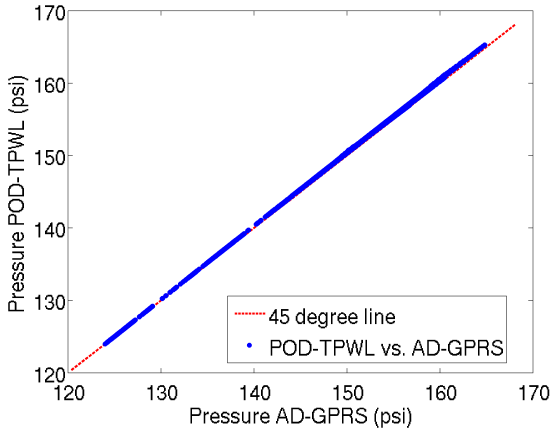
location (as opposed to the scheme described in Section 2.2.2, which does not explicitly consider this quantity).

Finally, we present cross plots comparing POD-TPWL and AD-GPRS results for the states (p and z_g in every block) at two different times. These results are shown, at 4000 and 6000 days, in Fig. 21. The pressure results are extremely accurate, while the overall molar fraction results display some scatter, but nonetheless retain a reasonable level of accuracy.

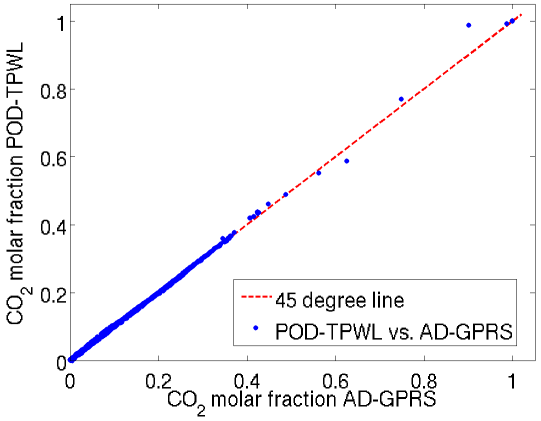
For this example, the run times for the AD-GPRS and POD-TPWL models are 16 minutes and 2.5 seconds, respectively. Thus POD-TPWL provides a speedup of about a factor of 380 in this case.



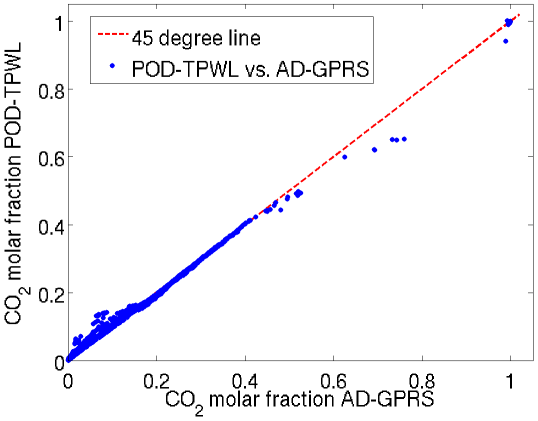
(a) Pressure at 4000 days



(b) Pressure at 6000 days



(c) CO_2 molar fraction at 4000 days



(d) CO_2 molar fraction at 6000 days

Figure 21: Comparison of pressure and CO_2 molar fraction between POD-TPWL and full-order reference solutions (Model 2)

3.3 Summary

In this section, we presented POD-TPWL results for two example cases involving horizontal CO_2 injection wells. The test cases entailed both time-varying BHP and injection rate schedules. POD-TPWL results were shown to provide reasonable accuracy relative to reference (full-order) AD-GPRS results, though POD-TPWL model error was seen to increase as the deviation of the test case from the training case increased. It is possible that some of the errors in the POD-TPWL models can be reduced through use of a more sophisticated point selection scheme. Runtime speedups observed using POD-TPWL (relative to the full-order simulations) were around a factor of 370.

4 Geological Perturbation

In this section, we will develop POD-TPWL models in which the perturbed 'control' variables are not well parameters, but are instead geological parameters. This general problem was considered by He et al. [42] within the context of oil-water reservoir simulation, and our formulation here follows that work. There are however some important differences between the development in [42] and the implementation here, particularly the fact that Galerkin projection was used for constraint reduction in [42], while here we use the Petrov-Galerkin procedure described in Section 2.

This work is in a relatively early stage, so the results presented in this section should be viewed as somewhat preliminary.

4.1 POD-TPWL Formulation

Consistent with the notation in Section 2, the set of discretized nonlinear algebraic equations is now expressed as:

$$\mathbf{g}(\mathbf{x}^{n+1}, \mathbf{x}^n, \boldsymbol{\gamma}) = \mathbf{0}. \tag{27}$$

Here, in place of the well control parameter $\mathbf{u}$ appearing in Eq. 3, is the geological parameter $\boldsymbol{\gamma}$. All other variables have the same definitions as in Section 2. The geological parameters used in the training run are designated $\boldsymbol{\gamma}_\omega$. The goal is now to generate results for test runs with different sets of geological parameters, which are denoted by $\boldsymbol{\gamma}$.

Consistent with Eq. 4, we now express new (test) solutions in terms of a Taylor series expansion, truncated at first order:

$$\mathbf{g}^{n+1} = \mathbf{0} \approx \mathbf{g}^{i+1} + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}}(\mathbf{x}^{n+1} - \mathbf{x}^{i+1}) + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i}}(\mathbf{x}^{n} - \mathbf{x}^{i}) + \frac{\partial \mathbf{g}^{i+1}}{\partial \boldsymbol{\gamma}_\omega}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega), \tag{28}$$

where $\mathbf{g}^{i+1} = \mathbf{g}(\mathbf{x}^{i+1}, \mathbf{x}^i, \boldsymbol{\gamma}_\omega) = \mathbf{0}$ and $\mathbf{g}^{n+1} = \mathbf{g}(\mathbf{x}^{n+1}, \mathbf{x}^n, \boldsymbol{\gamma}) = \mathbf{0}$, as noted in Section 2. Rearranging, we have

$$\mathbf{J}^{i+1}(\mathbf{x}^{n+1} - \mathbf{x}^{i+1}) = -[\mathbf{A}^{i+1}(\mathbf{x}^n - \mathbf{x}^i) + \mathbf{B}^{i+1}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega)], \tag{29}$$

where

$$\mathbf{J}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}}, \quad \mathbf{A}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i}}, \quad \mathbf{B}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \boldsymbol{\gamma}_\omega}. \tag{30}$$

Following the introduction of the POD representation ($\mathbf{x} = \boldsymbol{\Phi}\boldsymbol{\xi}$) and Petrov-Galerkin constraint reduction ($\boldsymbol{\Psi}^{i+1} = \mathbf{J}^{i+1}\boldsymbol{\Phi}$), the POD-TPWL model is expressed as:

$$\boldsymbol{\xi}^{n+1} = \boldsymbol{\xi}^{i+1} - (\mathbf{J}_r^{i+1})^{-1}[\mathbf{A}_r^{i+1}(\boldsymbol{\xi}^n - \boldsymbol{\xi}^i) + \mathbf{B}_r^{i+1}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega)], \tag{31}$$

where

$$\mathbf{J}_r^{i+1} = \boldsymbol{\Psi}^T \mathbf{J}^{i+1} \boldsymbol{\Phi}, \quad \mathbf{A}_r^{i+1} = \boldsymbol{\Psi}^T \mathbf{A}^{i+1} \boldsymbol{\Phi}, \quad \mathbf{B}_r^{i+1} = \boldsymbol{\Psi}^T \mathbf{B}^{i+1}. \tag{32}$$

We now consider the representation of the geological parameters. Following [42], we define $\boldsymbol{\gamma}$ to be log-transmissibility; i.e., $\boldsymbol{\gamma} = \log \mathbf{T}$, where $\mathbf{T}$ is the vector (of dimension n_T) containing all of the directional transmissibilities. Note that transmissibilities are associated with block-to-block interfaces, so n_T differs from the number of grid blocks. The term involving the geological parameters in the POD-TPWL representation is thus given by:

$$\mathbf{B}^{i+1}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega) = \frac{\partial \mathbf{g}^{i+1}}{\partial \boldsymbol{\gamma}_\omega}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega) = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{T}_\omega}\mathbf{D}_{T_\omega}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega), \tag{33}$$

where $\mathbf{T}_\omega$ designates the vector of transmissibilities for the training run and $\mathbf{D}_{T_\omega}$ is a diagonal matrix whose diagonal elements coincide with $\mathbf{T}_\omega$. Using Eq. 33, Eq. 31 can now be written as:

$$\boldsymbol{\xi}^{n+1} = \boldsymbol{\xi}^{i+1} - (\mathbf{J}_r^{i+1})^{-1}[\mathbf{A}_r^{i+1}(\boldsymbol{\xi}^n - \boldsymbol{\xi}^i) + \tilde{\mathbf{B}}_r^{i+1}(\boldsymbol{\gamma} - \boldsymbol{\gamma}_\omega)], \tag{34}$$

with

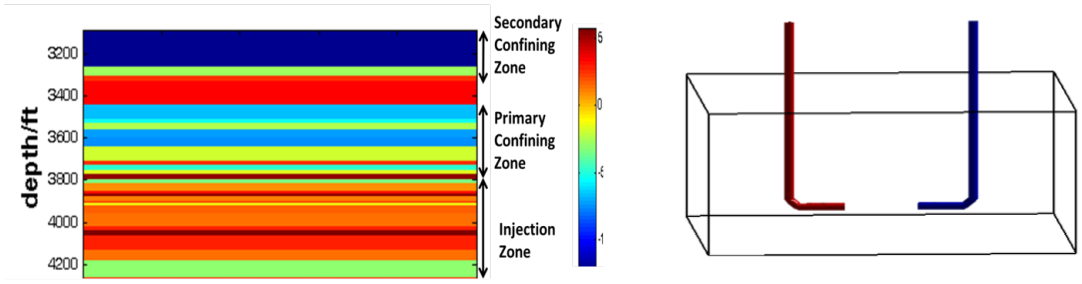
$$\tilde{\mathbf{B}}_r^{i+1} = \boldsymbol{\Psi}^T \frac{\partial \mathbf{g}^{i+1}}{\partial \boldsymbol{\gamma}_\omega}\mathbf{D}_{T_\omega}. \tag{35}$$

This defines the POD-TPWL model with geological control parameters. Point selection is accomplished as described in Section 2.2.2.

4.2 Problem Set Up

We test this procedure on a vertical cross-section of the purely-layered Mount Simon model considered in Section 3. The storage aquifer is represented on a 30×30 grid (see Fig. 22a; this represents the training geology). The full regional system contains 46×30 blocks. We again set $k_z = 0.1k_x$ in the training run. Gravitational effects are neglected in these runs (due to a current limitation in the linkage between POD-TPWL with geological perturbations and AD-GPRS).

CO_2 injection is from two horizontal wells, as shown in Fig. 22b, which are each of length 1575 ft. Both wells inject CO_2 with time-varying rate controls for 3000 days. The cumulative CO_2 injected at the end of 3000 days is about 1.3% of the pore volume of the storage aquifer. The rate controls at surface conditions (293 K, 1.013 bar) for both wells are shown in Fig. 23. Note that the rate schedule is the same for both the training and test cases. The other model properties are the



(a) Horizontal permeability for training case (b) Well locations

Figure 22: Permeability field (in $\log k$) and well locations for geological perturbation example

same as in the Mount Simon example in Section 3.

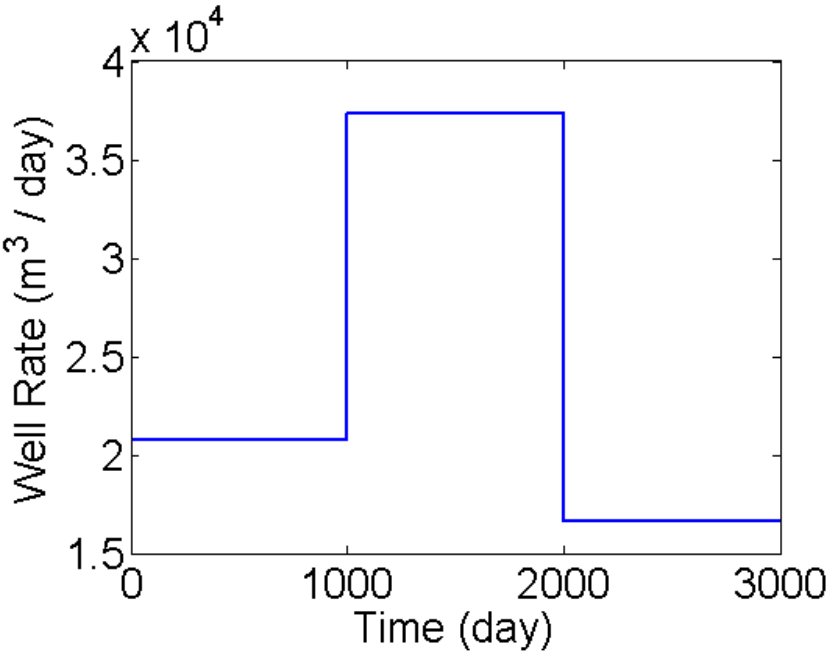


Figure 23: Injection rates in training and test cases (rates are the same for both wells)

The POD basis matrix is constructed from 80 snapshots generated during two full-order training runs. The AD-GPRS full-order system for this case contains 2760 primary variables (2×1380). In the POD-TPWL model, we use $l_p = l_z = 37$. The control variable $\boldsymbol{\gamma}$ in Eq. 34 in this case is also of dimension 2760. Theoretically, this dimension should be the number of block-to-block connections in the system, which is $(46-1) \times 30 + 46 \times (30-1) = 2684$. However, to simplify the linkage with AD-GPRS, we introduce a small amount of redundancy in the data. In all of the test cases, we keep the well index equal to its value in the training simulation. This is meant to reflect the fact that the geology is reasonably certain in regions intersected by wells.

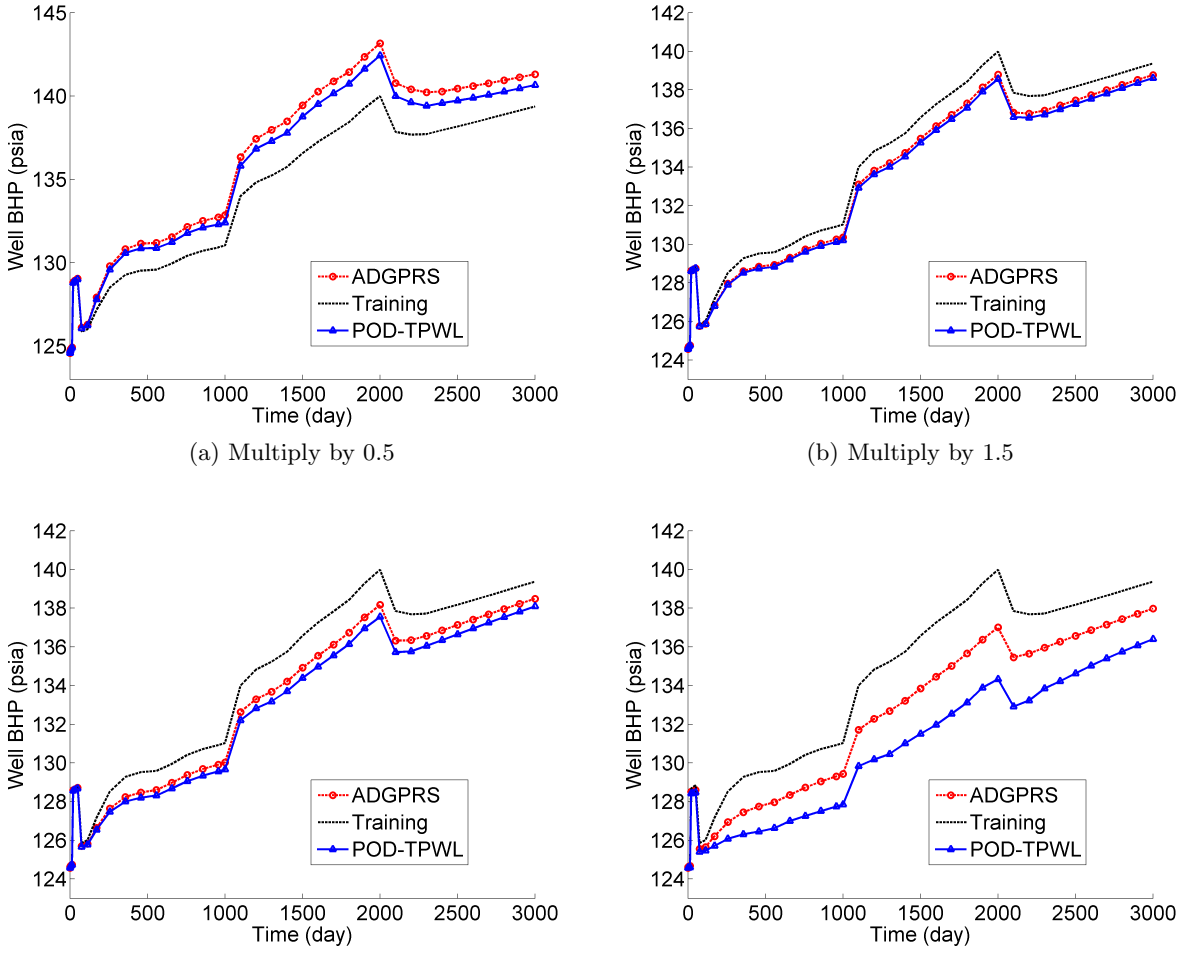
4.3 POD-TPWL Results

We now present POD-TPWL results for test cases with different types of geological perturbations. Three different examples are considered, in which we perturb, relative to the training case (1) the transmissibilities of all interfaces in the model, (2) only the vertical transmissibilities, and (3) the transmissibilities in a few selected layers. Because the wells are symmetrically located within the model and the injection schedules for the two wells are identical (and because the model is completely layered), both wells display identical time-varying BHPs. Therefore we present results only for Well 1.

In the first set of test cases, we multiply all permeabilities (and thus all transmissibilities) by constant factors (0.5, 1.5, 2.0, 5.0) relative to the training case. BHP results for these runs are shown in Fig. 24. The various curves are as defined in Section 3. The test-case BHPs differ from the training-run BHPs by very little in these runs – typically by only a few psi or less. This may be because the well indices are the same in the test cases as in the training run. The POD-TPWL model does, however, provide results that capture the basic trends, though error is apparent for the case with the largest perturbation (Fig. 24d).

We next present results for a test case in which the vertical transmissibilities are all multiplied by 0.1 relative to their values in the training case. Injection well BHP results for this run are shown in Fig. 25. We again see only small differences between training and test results. The POD-TPWL model captures the general trend of the AD-GPRS results, though some error is apparent between 1500 and 2000 days.

In our final test case, the transmissibilities in layers 23 to 27 are all multiplied by a factor of 10. These layers can be viewed as key layers because the wells are completed in layer 25. Results for this case are shown in Fig. 26. Consistent with previous examples, we observe generally accurate POD-TPWL results for time-varying BHP, though the training- and test-case results are again quite close.



(a) Multiply by 0.5

(b) Multiply by 1.5

(c) Multiply by 2.0

(d) Multiply by 5.0

Figure 24: Injection well BHPs for test cases with all transmissibilities perturbed by constant factors (Well 1)

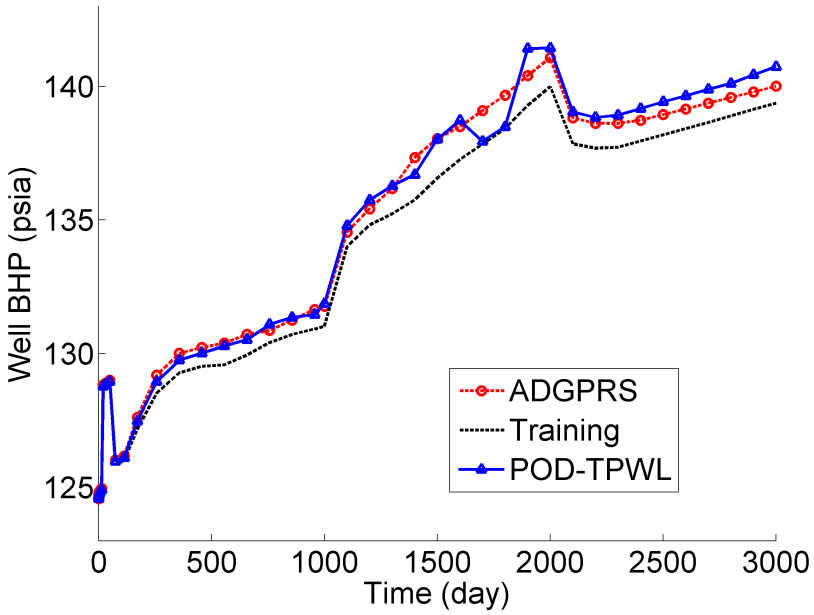


Figure 25: Injection well BHPs for test case with perturbed vertical transmissibilities (Well 1)

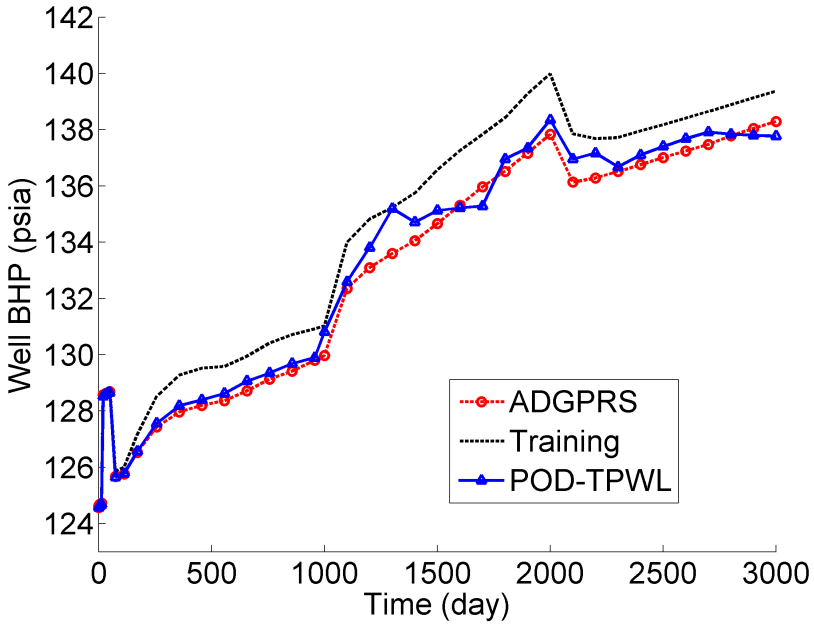


Figure 26: Injection well BHPs for test case with perturbed transmissibilities in layers 23–27 (Well 1)

4.4 Summary

In this section, we extended the POD-TPWL model to cases involving geological perturbations. In the test cases presented, there were only small differences between the training and test-case simulation results for CO_2 injection well BHPs, though the test-case BHPs were accurately captured by the POD-TPWL model in most instances. The case that displayed the least accurate POD-TPWL results involved the use of a global permeability/transmissibility multiplier of 5.0, which is relatively large. The runtime speedups for the cases in this section were modest compared with those achieved in Section 3. Specifically, here we obtained speedup factors of only about 50 (from around 2 minutes for full-order AD-GPRS runs to 2-3 seconds for POD-TPWL). The decreased speedup may be due in part to the fact that the full-order model is fairly simple. Greater speedups will likely be achieved if larger and more complicated models are tested, especially if we use a Karhunen-Loève representation of transmissibility, as in [42]. The Karhunen-Loève representation enables geological models to be defined in terms of a reduced set of parameters, and is analogous to the POD-based representation used for the states (i.e., $\mathbf{x} = \mathbf{\Phi}\boldsymbol{\xi}$).

The results presented in this section are preliminary, though they do suggest that the basic POD-TPWL model for geological perturbation has been implemented (essentially) correctly. Further development and testing of this capability will be the subject of future work.

5 Summary and Conclusions

In this report, a compositional POD-TPWL reduced-order model was presented for CO_2 sequestration problems. This work built on an earlier POD-TPWL formulation, which considered oil-gas compositional systems within a reservoir simulation setting [39, 40]. POD-TPWL techniques entail the representation of states using a POD-based procedure and the treatment of nonlinearity using linearization around previously simulated (training) solutions. The set of governing equations is projected into an appropriate subspace using a constraint reduction procedure. In this work, consistent with [39, 40], a Petrov-Galerkin treatment was used for this purpose.

New features introduced into the POD-TPWL formulation presented here include the use of rate-controlled wells (rather than BHP-controlled wells, as have been used in previous implementations) and horizontal wells (in contrast to previous POD-TPWL models which only used vertical wells). The implementation with rate controls is much more involved than with BHP controls as it requires manipulation of AD-GPRS matrices in order to provide the derivatives used in the POD-TPWL model. Rate control is important for CO_2 storage problems since we typically have a target volume of CO_2 to inject at each time step. A prototype implementation involving the use of geological parameters as the control parameters was also introduced. This model is similar to that in [42] except here we used a Petrov-Galerkin constraint reduction procedure (in contrast to the Galerkin procedure in [42]).

POD-TPWL results for BHP- and rate-controlled test runs (test runs differ from the training cases used to construct the POD-TPWL model) were presented for two example cases — a

synthetic channelized aquifer model and a simplified Mount Simon model. The POD-TPWL solutions typically showed reasonably close agreement with full-order (AD-GPRS) reference solutions. POD-TPWL model error was seen to increase as the difference between the training- and test-case controls increased, as would be expected since the method is based on a linearization procedure.

The computational times associated with the full-order (AD-GPRS) and POD-TPWL solutions, as well as the time required for POD-TPWL model construction, are shown in Table 1. Results are presented for the three CO_2 storage cases considered (timings are very similar for BHP- and rate-controlled runs). The runtime speedup for POD-TPWL relative to AD-GPRS is also shown. We see that POD-TPWL model construction requires 0.17 and 0.35 of the time required for one AD-GPRS simulation for Models 1 and 2 (a smaller fraction of AD-GPRS runtime is required for the geological perturbation case). The computations entailed in POD-TPWL model construction include loading the derivative matrices and snapshots, performing SVDs of the snapshot matrices, computing reduced matrices (e.g., $\mathbf{J}_r^{i+1} = (\mathbf{\Psi}^{i+1})^T \mathbf{J}^{i+1} \mathbf{\Phi}$), etc.

Runtime speedups of about a factor of 370 were observed for Models 1 and 2, which highlights the benefit of solving low-order linear problems rather than high-order nonlinear problems. Less dramatic speedup was achieved in the runs with geological perturbations, but those results do suggest that the basic functionality has been implemented (essentially) correctly. We note that, for more complicated multicomponent simulations, such as those presented in Appendix B, the POD-TPWL model construction time is longer – nearly the time required for one full-order simulation run.

Table 1: Timings for various modeling components (in seconds)

	Model 1	Model 2	Geol. Pert. Case
AD-GPRS runtime	1080	960	100
POD-TPWL model construction	180	340	15
POD-TPWL runtime	3	2.5	2
POD-TPWL runtime speedup	360	384	50

There are a number of directions that should be considered in future work. As noted in Section 3, the point selection procedure used in this study displays limitations in some cases, and a more general treatment should be developed. It will also be useful to test the relative performance of POD-TPWL models, in terms of their ability to capture plume fronts, with different point selection schemes. As indicated in Section 4, geological models can be represented compactly, and the use of such representations in the POD-TPWL model with geological perturbations should be further explored. This was considered in [42], but not for CO_2 storage problems. A recent method, referred to as optimization-based principal component analysis [71], enables the approximate (concise) representation of non-Gaussian permeability fields, and this approach should be incorporated into our POD-TPWL model. The POD-TPWL model should also be applied for optimizations of the type considered in [13]. This will require the development of appropriate retraining strategies since, as the optimization proceeds, the controls will eventually differ considerably from those used

in the training run. Finally, it may be useful to consider other numerical reduced-order methods such as POD-DEIM (e.g., [36]) for CO_2 storage problems.

Acknowledgments

We are grateful to Srikanta Mishra (Battelle), the PI of the overall project, for many useful discussions and suggestions, and to Alain Bonneville (Pacific Northwest National Laboratory) for providing the Mount Simon data and model used in this work. We also thank Oleg Volkov and Vladislav Bukshtynov (Stanford) for their assistance with AD-GPRS.

A Constraint Reduction for POD-TPWL

This Appendix presents a detailed assessment of POD-TPWL constraint reduction procedures in oil-gas and oil-water models. It corresponds to a paper that has been accepted for publication in *International Journal for Numerical Methods in Engineering* (doi:10.1002/nme.4874). Because this Appendix was written as a separate paper, there is some overlap between the text and equations here and that in the body of this report (there are also some slight stylistic differences). As noted earlier, within the body of the report, the work in Appendix A is referenced as He and Durlofsky [40].

A.1 Summary of Appendix A

The properties and numerical performance of reduced-order models based on trajectory piecewise linearization (TPWL) and proper orthogonal decomposition (POD) are assessed. The target application is subsurface flow modeling, though our findings should be applicable to a range of problems. The errors arising at each step in the POD-TPWL procedure are described. The impact of constraint reduction on accuracy and stability is considered in detail. Constraint reduction entails projection of the overdetermined system into a low-dimensional subspace, in which the system is solvable. Optimality conditions for constraint reduction, in terms of error minimization, are derived. Galerkin and Petrov-Galerkin projections are shown to correspond to optimality in norms that involve weighting with the Jacobian matrix. Two new treatments, inverse projection and weighted inverse projection, are suggested. These methods minimize error in appropriate norms, though they require substantial preprocessing computations. Numerical results are presented for oil reservoir simulation problems. Galerkin projection provides reasonable accuracy for simpler oil-water systems, though it becomes unstable in more challenging cases. Petrov-Galerkin projection is observed to behave stably in all cases considered. Weighted inverse projection also behaves stably, and it provides the highest accuracy. Runtime speedups of 150–400 are achieved using these POD-TPWL models.

A.2 Introduction

The development of reduced-order modeling procedures for nonlinear problems is a topic of great interest in many application areas. The issues and approaches considered in this Appendix are relevant to a wide range of problems, though our focus here is on subsurface flow modeling — specifically oil reservoir simulation. Within that setting, detailed finite-volume-based flow simulators, which track the movement of multiple components in multiple phases through porous subsurface formations, are typically used to model the production of oil and gas. Important applications within this area, such as production optimization, uncertainty assessment and data assimilation, require large numbers of simulation runs. These applications, like many in other engineering fields, are extremely demanding computationally using standard full-order simulations, and they could benefit greatly from the use of fast, accurate and robust reduced-order models (ROMs).

Essentially, the ROMs considered in this work include three key components: state reduction,

nonlinearity treatment, and constraint reduction. State reduction entails the expression of full-order states (i.e., the vector of state variables in all grid blocks in the model) in terms of a small set of reduced variables. Nonlinearity treatment involves the approximate representation of nonlinear effects. Approaches include the construction of approximate/reduced nonlinear terms or Jacobian matrices, and/or the use of some type of (piecewise) linearization procedure. Constraint reduction, which is the focus of this work, is required because, after the introduction of state reduction, there are many more equations than unknowns. It is the constraint reduction matrix that defines the low-dimensional subspace in which the residue of the original system is driven to zero. As we will see, the choice of this matrix can have a large impact on the accuracy and stability of the resulting ROM.

The use of state reduction is based on the assumption that the state vectors of the full-order system essentially lie in a lower-dimensional subspace. This is often a reasonable assumption because the states that can arise are defined through initial conditions and system dynamics, which are not infinitely variable. With this assumption, a basis for the subspace, which projects the full-order (high-fidelity) state into a low-order representation, can be constructed. In many ROM procedures, including the one considered here, the state reduction basis matrix $\mathbf{\Phi}$ is constructed through use of proper orthogonal decomposition (POD). With this approach, a data matrix, containing as its columns 'snapshots' (solution vectors) computed during 'training' simulations, is first constructed. The left singular vectors of the singular value decomposition of the data matrix define the columns of the basis matrix $\mathbf{\Phi}$. POD-based ROMs have been used in a number of application areas [9, 12, 47, 66, 70], including subsurface flow simulation [18, 67]. Other approaches, such as balanced truncation [28, 43, 49] and Krylov subspace methods [32, 68, 69, 77], have also been successfully applied.

State reduction procedures decrease the number of unknowns that must be determined at each time step in a dynamic simulation. However, for nonlinear time-variant problems, the speedups achieved through the use of state reduction alone are typically quite modest. Specifically, for reservoir simulation problems, speedup factors of at most 10 have been achieved using this approach [18, 67]. This is because some of the order-reduction computations have a computational complexity that scales with the dimension of the full-order problem. If such computations are performed at each (nonlinear) iteration at every time step, as is the case in [18, 67], the observed speedup will be limited.

Various treatments have been proposed to further accelerate ROMs for nonlinear problems. These include the discrete empirical interpolation method (DEIM) and trajectory piecewise linearization (TPWL). DEIM, first proposed by Chaturantabut and Sorensen [21, 22], reduces the dimension of nonlinear functions in the governing partial differential equation (PDE) using an empirically derived basis. During the inline (runtime) stage, reduced-order nonlinear functions are determined through computations involving only a small number of grid blocks, which greatly reduces inline computational demands. Carlberg *et al.* [19] further extended the method to treat nonlinear algebraic systems obtained from application of Newton's method. They applied a com-

pressive tensor approximation to enable the fast construction of reduced Jacobian matrices, which were then used in inline computations. DEIM has been applied successfully for different applications [19, 22, 44], though its implementation does require nonlinear terms to be evaluated at particular grid blocks during inline processing. This is intrusive with respect to the full-order simulator, which could pose a problem in the use of DEIM with general purpose reservoir simulators. We note finally that a prototype DEIM procedure has been developed for reservoir flow [37], though only small two-dimensional models have thus far been considered.

Trajectory piecewise linearization (TPWL), proposed by Rewienski and White [62], handles nonlinearity by constructing local (piecewise) linearizations around previously simulated (training) solutions. Because new (test) runs entail linearization around training 'points,' the order reduction computations can all be performed offline (i.e., in a preprocessing step). Thus the inline computations involve only low-dimensional linear solutions. TPWL has been combined with POD and applied for a number of subsurface flow problems. These include oil-water models [16, 17, 41], idealized thermal simulation cases [63], compositional systems [39], and ensemble-based data assimilation [42]. Construction of the POD-TPWL model for reservoir simulation problems requires preprocessing (offline) computations equivalent to about 3–4 full-order simulations, though runtime speedups of 200–1000 were reported in the studies noted above.

As indicated earlier, constraint reduction entails the projection of the overdetermined system into a low-dimensional subspace in which the residue is driven to zero. This subspace is defined by the constraint reduction matrix $\mathbf{\Psi}$. For ROMs based on Krylov subspace or balanced truncation (including balanced POD [46, 74], in which POD is used to approximate the Gramians in the balanced truncation method), the appropriate constraint reduction matrix is provided from theory [38]. For POD-based methods, there is no unambiguous choice for $\mathbf{\Psi}$, and different approaches have been used.

In the initial POD-TPWL method for reservoir simulation [16], a Galerkin projection scheme [4], in which $\mathbf{\Psi} = \mathbf{\Phi}$ (recall that $\mathbf{\Phi}$ is the dimension-reduction matrix), was applied. However, as shown in [41], this approach can lead to numerical stability problems in some cases. In [41], a procedure was devised to select the columns in $\mathbf{\Phi}$ to improve system stability. This approach was shown to perform well for the oil-water cases considered, but it does not guarantee stability. Bond and Daniel [5] proposed that the constraint reduction matrix $\mathbf{\Psi}$ be designed to guarantee the stability of the reduced system through satisfaction of Lyapunov stability criteria. However, if only stability is considered, the reduced-order model may be inaccurate. If accuracy is also taken into account, a matrix optimization problem must be solved to obtain the optimal $\mathbf{\Psi}$, and this is very expensive for large systems. A Petrov-Galerkin projection scheme was recently used for POD-based DEIM by Carlberg *et al.* [19]. This approach provided numerical stability at reasonable computational cost (though stability is still not guaranteed). A Petrov-Galerkin procedure was also used in [10] for linear model reduction, and recently in [39] for POD-TPWL compositional reservoir simulation. This approach has not, however, been studied systematically within the context of POD-TPWL.

In this work we assess the accuracy and stability of various constraint reduction treatments for

POD-TPWL models. The approaches presented should be relevant for POD-TPWL procedures in a range of application areas, though our implementation and numerical results are for subsurface flow problems. Following a brief assessment of the POD-TPWL errors that arise from state reduction and linearization, we discuss the characteristics of several constraint reduction procedures. Optimality conditions for these approaches, which are based on error minimization, are presented. The methods considered include Galerkin projection, Petrov-Galerkin projection, and two new methods, inverse projection and weighted inverse projection. We also provide linear stability criteria for POD-TPWL models. The numerical accuracy and stability of the different constraint reduction procedures are compared for oil-water and oil-gas compositional flow examples. Our results demonstrate the relative advantages of the different approaches and suggest directions for future research.

This Appendix proceeds as follows. In Section A.3 we briefly discuss the reservoir simulation problems targeted in this work. In Section A.4 the POD-TPWL model is derived, and the error incurred at each step is discussed. Optimal constraint reduction procedures are derived in Section A.5, and stability requirements are discussed in Section A.6. In Section A.7, the performance of Galerkin projection and Petrov-Galerkin projection are investigated in detail for three test cases. The two new constraint reduction methods, inverse projection and weighted inverse projection, are developed in Section A.8. Numerical results for these approaches are also presented. A summary of our findings and suggestions for future work are provided in Section A.9.

A.3 Problem Description

Our specific interest here is in the simulation of oil-water and oil-gas compositional systems. Oil-water systems are commonly used to model oil production driven by the injection of water (referred to as waterflooding), while compositional systems are used to model enhanced oil recovery processes, which often involve the injection of gas, as well as CO_2 storage operations. Our descriptions here are brief; for more details see, e.g., [3, 16, 27, 35, 38, 72, 78].

The governing equations for oil-water systems consist of statements of mass conservation for oil and water, combined with Darcy's law, which relates the flow of each fluid phase to the pressure gradient. These equations include accumulation, flux, and source/sink terms, and can be written as:

$$\frac{\partial}{\partial t}\left(\phi\rho_j S_j\right) - \nabla\cdot\left[\rho_j\lambda_j\mathbf{k}\left(\nabla p_j - \rho_j g\nabla D\right)\right] + q_j^w = 0, \quad j = o,\ w, \tag{36}$$

where the subscript j designates the fluid phase (o indicates oil and w water). Here t is time, ϕ is porosity (volume fraction of the pore space), ρ_j is the phase density, S_j is the phase saturation (volume fraction of phase j within the pore space) , λ_j is the phase mobility, which is typically a nonlinear function of S_j, $\mathbf{k}$ is the absolute permeability tensor ($\mathbf{k}$ is essentially a flow conductivity and is a property of the rock), p_j is the phase pressure, g is the gravitational acceleration, D is the depth, and q_j^w is the source/sink term (the superscript w indicates that this term is driven by wells). The system is closed by adding the saturation constraint ($S_o + S_w = 1$) and the capillary

pressure relationship $p_c(S_w) = p_o - p_w$. The primary unknowns in Equation 36 are often taken to be the oil phase pressure p_o and water phase saturation S_w. Other quantities (p_w and S_o) can be easily computed block-by-block once p_o and S_w are determined. See [16] for more details on the oil-water problem formulation in the context of POD-TPWL. Note that equations of the form of Equation 36 arise in many problems involving flow and transport.

For compositional systems, we track a total of n_c components (as opposed to two components in oil-water systems). The governing equations for oil-gas compositional systems resemble Equation 36, in that they entail statements of mass conservation for each component and Darcy's law for each phase, though they are complicated by the fact that components partition between the oil and gas phases. Phase equilibrium equations for each component are therefore additionally required to determine the fraction of each component in each phase. For isothermal compositional systems with n_c components in two phases, there are a total of $2n_c + 4$ unknown variables in each grid block [39]. Practical systems are typically modeled with ∼4–10 components, so computational demands for large models can be substantial.

In compositional models there are, however, only n_c primary equations and n_c primary unknowns for each grid block. This set of equations must be solved as a fully-coupled system. The remaining $n_c + 4$ unknowns decouple and can be solved block-by-block. The n_c primary equations are typically the mass conservation equations [27, 72, 78]. Various choices for the n_c primary variables are possible. Most common is the so-called natural formulation, in which the primary variables consist of the oil phase pressure p_o and $n_c - 1$ phase-dependent variables (such as the mole fraction of component c in the gas phase). The natural formulation, however, requires variable switching when a phase disappears. This introduces complications in the context of reduced-order modeling since it is much more straightforward for the ROM procedure to treat the same types of variables in the test and training simulations. We thus apply the less commonly used molar formulation [72], in which the primary variables are p_o and $n_c - 1$ overall mole fractions. These quantities are well defined at all times in all grid blocks, so variable switching is not required.

The governing equations and detailed discretizations differ for oil-water and compositional problems. Nonetheless, the fully-implicit discretized representations, for a wide range of problems including these, can be written as a general set of nonlinear algebraic equations in the following form:

$$\mathbf{g}^{n+1} = \mathbf{g}\left(\mathbf{x}^{n+1}, \mathbf{x}^{n}, \mathbf{u}^{n+1}\right) = \mathbf{0}. \tag{37}$$

Here $\mathbf{g}$ is the residual vector we seek to drive to zero, n and $n+1$ denote time level, $\mathbf{u}$ is the set of control parameters, and $\mathbf{x}$ designates the state vector (primary variables in each grid block). We denote the number of grid blocks as n_b and the dimension of the state vector $\mathbf{x}$ (that is, the total number of primary variables) as n_v. In oil-water systems, $\mathbf{x}$ contains oil pressure p (from here on we use p in place of p_o) and water saturation S_w in each grid block, so $n_v = 2n_b$. In compositional systems, $\mathbf{x}$ contains pressure p and the overall mole fraction, designated z_c, for $n_c - 1$ components. In this case, $n_v = n_c n_b$. In Equation 37, $\mathbf{u}$ is the set of specified control parameters that drive the oil recovery process. These are taken here to be the pressures of injection or production wells

(referred to as bottom-hole pressures or BHPs), though they could also be injection or production flow rates. In either case, these terms enter through the source terms in the governing equations (e.g., q_j^w in Equation 36) and thus strongly impact the numerical solutions. In all cases, $\mathbf{x}^n$ is known from the previous time step or the initial condition, and the goal is to compute $\mathbf{x}^{n+1}$.

In the full-order simulation, Equation 37 is solved using Newton's method. This entails, at each iteration, the solution of the high-dimensional linear system

$$\mathbf{J}\boldsymbol{\delta} = -\mathbf{g}, \tag{38}$$

where $\mathbf{J}$ is the Jacobian matrix, given by $\mathbf{J} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ evaluated at the current estimate of $\mathbf{x}^{n+1}$, and $\boldsymbol{\delta} = \mathbf{x}^{n+1,\nu+1} - \mathbf{x}^{n+1,\nu}$ is the update vector, where ν designates iteration. Convergence is achieved once an appropriate norm of $\mathbf{g}^{n+1}$ is less than a specified tolerance, and the solution states are then designated $\mathbf{x}^{n+1}$.

For practical reservoir simulation problems, $n_v \sim O(10^4 - 10^6)$. In addition, the high degree of nonlinearity of Equation 37 can result in substantial numbers of Newton iterations, small time steps, and frequent time-step cuts. The combination of nonlinearity and high dimensionality leads to very large computational demands, especially when thousands or tens of thousands of simulations must be performed, as may be the case for production optimization computations. The POD-TPWL approach we now describe can provide a much more efficient (though approximate) solution of Equation 37.

A.4 POD-TPWL Model and Assessment of Error

In this section we will consider general POD-TPWL models, applicable to a wide range of systems. We will assess the errors incurred at each step of the POD-TPWL procedure. These include linearization error, state reduction error, constraint reduction error, and error propagated from the previous time step (which we relate to stability). Most aspects of this discussion are quite general, though some are specific to the particular models and fluid systems under consideration. Detailed derivations of POD-TPWL models for oil-water [16, 41] and compositional systems [39] have been presented previously and should be consulted for more details.

Error in reduced-order models has been analyzed by a number of investigators. Rathinam and Petzold [59], for example, presented an error analysis for POD-based reduced-order ODE systems. Chaturantabut and Sorensen [23] provided a state-space error estimate for nonlinear model reduction based on POD-DEIM. Our discussion here will be focused on POD-TPWL.

In the following discussion, the POD-TPWL equations are derived from the original system of equations (Equation 37) by introducing a series of approximations. Solutions associated with different levels of approximation will be denoted $\mathbf{x}_\alpha^{n+1}$ ($\alpha = 1, 2, 3, 4$), with increasing α indicating a more approximate solution. Consistent with this, we denote the exact solution to the full set of nonlinear algebraic equations (Equation 37) as $\mathbf{x}_0^{n+1}$. The solution that contains linearization error is designated $\mathbf{x}_1^{n+1}$, the solution that contains linearization error and state reduction error

is denoted $\mathbf{x}_2^{n+1}$, the solution that contains linearization, state reduction and constraint reduction error is designated $\mathbf{x}_3^{n+1}$, and the solution that contains all of these errors plus error propagated from the previous time step is denoted $\mathbf{x}_4^{n+1}$.

A.4.1 Trajectory Piecewise Linearization

In order to construct the POD-TPWL model, we must first perform one or more full-order 'training' simulations for some specific control parameters, which we designate $\mathbf{u}^{i+1}$. This corresponds to generating solutions to the following equation:

$$\mathbf{g}^{i+1} = \mathbf{g}(\mathbf{x}^{i+1}, \mathbf{x}^i, \mathbf{u}^{i+1}) = \mathbf{0}, \quad i = 0, \ldots, n_t - 1, \tag{39}$$

where n_t is the number of time steps. Note that we use superscripts i and $i+1$ to indicate sequential 'points' (in time) in a training simulation; i.e., $\mathbf{x}^i$ and $\mathbf{x}^{i+1}$ are the solutions of the training simulation at time steps i and $i+1$.

In order to construct the solution for a new set of controls (designated $\mathbf{u}^{n+1}$), rather than solve Equation 37 iteratively using Newton's method, we instead represent the new residual vector $\mathbf{g}^{n+1}$ in terms of a Taylor series expansion around the training solution. We refer to the new simulation as a 'test' simulation. Neglecting higher-order terms, we write:

$$\mathbf{g}^{n+1} = \mathbf{0} \approx \mathbf{g}^{i+1} + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}} \left(\mathbf{x}^{n+1} - \mathbf{x}^{i+1}\right) + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i}} \left(\mathbf{x}^{n} - \mathbf{x}^{i}\right) + \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{u}^{i+1}} \left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right). \tag{40}$$

Here $\mathbf{x}^n$ indicates the solution at the previous time level in the test simulation, $\mathbf{x}^{n+1}$ indicates the test solution that we wish to compute, and $\mathbf{x}^i$ and $\mathbf{x}^{i+1}$ are sequential solutions in the training simulation.

From Equation 39 we know that $\mathbf{g}^{i+1} = \mathbf{0}$. This allows us to express Equation 40, after some rearrangement, as follows

$$\mathbf{J}^{i+1}\mathbf{x}^{n+1} = \mathbf{J}^{i+1}\mathbf{x}^{i+1} - \left[\mathbf{A}^{i+1}\left(\mathbf{x}_0^n - \mathbf{x}^i\right) + \mathbf{B}^{i+1}\left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right)\right], \tag{41}$$

where

$$\mathbf{J}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}}, \quad \mathbf{A}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i}}, \quad \mathbf{B}^{i+1} = \frac{\partial \mathbf{g}^{i+1}}{\partial \mathbf{u}^{i+1}}. \tag{42}$$

Here $\mathbf{J}^{i+1} \in \mathbb{R}^{n_v \times n_v}$ is the Jacobian matrix at time step $i+1$ (evaluated upon convergence) of the training simulation, $\mathbf{A}^{i+1} \in \mathbb{R}^{n_v \times n_v}$, and $\mathbf{B}^{i+1} \in \mathbb{R}^{n_v \times n_u}$, where n_u is the dimension of the control vector $\mathbf{u}$ (typically there are significantly fewer wells than grid blocks, so $n_u << n_v$). Note that $\mathbf{x}_0^n$, the exact solution of Equation 37 at the previous time step, appears in Equation 41. This is because the error we are now considering corresponds to the error incurred in a single time step. Later, in Section A.4.4, we will incorporate the error propagated from the previous time step into our analysis.

We denote the solution to Equation 41 as $\mathbf{x}_1^{n+1}$. This equation approximates the high-dimensional

nonlinear system in Equation 37 as a high-dimensional linear system. The (linearization) error incurred in this step is referred to as $\mathbf{e}_{01}^{n+1}$, where $\mathbf{e}_{01}^{n+1} = \mathbf{x}_0^{n+1} - \mathbf{x}_1^{n+1}$.

A.4.2 Proper Orthogonal Decomposition

Equation 41 is linear but it is still expressed in the high-dimensional space. To reduce the number of unknowns, we now apply POD. This enables us to write $\mathbf{x} = \mathbf{\Phi}\boldsymbol{\xi}$, where $\mathbf{\Phi} \in \mathbb{R}^{n_v \times l}$ is the basis matrix and $\boldsymbol{\xi} \in \mathbb{R}^{l \times 1}$ is the reduced state vector. Because $l << n_v$, the system state can be expressed in terms of a relatively small number of variables.

Proper orthogonal decomposition (POD) has been used in the context of reduced-order modeling by a number of researchers; see, e.g., [9, 12, 18, 47, 66, 67, 70]. In POD, the columns of the basis matrix $\mathbf{\Phi}$ are the leading singular vectors of the snapshot matrices $\mathbf{X}$. Snapshot matrices contain, as their columns, the solution vectors computed during one or more training simulations. In this work we typically use two or three training runs to provide a sufficient number of snapshots for the POD basis construction.

As discussed in [16] and [39], we apply POD separately to pressure snapshots and water saturation snapshots (in oil-water problems), or to pressure snapshots and overall mole fraction snapshots (in compositional problems). For oil-water systems we have

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{S}_w \end{bmatrix} \approx \mathbf{\Phi}\boldsymbol{\xi} = \begin{bmatrix} \mathbf{\Phi}_p & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_S \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi}_p \\ \boldsymbol{\xi}_S \end{bmatrix}, \tag{43}$$

where $\mathbf{\Phi}_p \in \mathbb{R}^{n_b \times l_p}$ and $\mathbf{\Phi}_S \in \mathbb{R}^{n_b \times l_S}$ are the basis matrices for pressure and water saturation respectively, $\boldsymbol{\xi}_p \in \mathbb{R}^{l_p \times 1}$ and $\boldsymbol{\xi}_S \in \mathbb{R}^{l_S \times 1}$ are the reduced state vectors for pressure and water saturation, and $\mathbf{\Phi} \in \mathbb{R}^{n_v \times l}$, where $l = l_p + l_S$, is the basis matrix for the entire state vector $\boldsymbol{\xi}$. Note that, in general, the number of columns in $\mathbf{\Phi}_p$ differs from that in $\mathbf{\Phi}_S$ (i.e., $l_p \neq l_S$).

Similarly, for compositional systems we have

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_{n_c-1} \end{bmatrix} \approx \mathbf{\Phi}\boldsymbol{\xi} = \begin{bmatrix} \mathbf{\Phi}_p & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_z \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi}_p \\ \boldsymbol{\xi}_z \end{bmatrix}, \tag{44}$$

where $\mathbf{\Phi}_z \in \mathbb{R}^{n_b(n_c-1) \times l_z}$ is the basis matrix for the overall mole fraction variables, $\boldsymbol{\xi}_z \in \mathbb{R}^{l_z \times 1}$ is the corresponding reduced state vector, and $\mathbf{\Phi} \in \mathbb{R}^{n_v \times l}$, where $l = l_p + l_z$, is the basis matrix for the entire state vector $\boldsymbol{\xi}$. Procedures for specifying l_p and l_S, or l_p and l_z, which can be based on 'energy' criteria or stability considerations (the latter are discussed in Section A.6 and illustrated in Section A.7.4) are described in [16, 39, 41]. We note finally that all columns of the submatrices in $\mathbf{\Phi}$, as well as the columns of the overall $\mathbf{\Phi}$ matrix, are orthonormal (meaning $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{I}$).

We now introduce the reduced representation $\mathbf{x} = \mathbf{\Phi}\boldsymbol{\xi}$ to the right hand side of Equation 41,

which gives

$$\mathbf{J}^{i+1}\mathbf{x}^{n+1} = \mathbf{J}^{i+1}\mathbf{\Phi}\boldsymbol{\xi}^{i+1} - \left[\mathbf{A}^{i+1}\mathbf{\Phi}\left(\boldsymbol{\xi}_0^n - \boldsymbol{\xi}^i\right) + \mathbf{B}^{i+1}\left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right)\right]. \tag{45}$$

The solution to Equation 45 is denoted $\mathbf{x}_2^{n+1}$. The additional error incurred in this step is due to state reduction and can be expressed as $\mathbf{e}_{12}^{n+1} = \mathbf{x}_1^{n+1} - \mathbf{x}_2^{n+1}$. Note that $\boldsymbol{\xi}_0^n$ is the reduced representation of $\mathbf{x}_0^n$, the true solution at time step n; i.e., $\boldsymbol{\xi}_0^n = \mathbf{\Phi}^T\mathbf{x}_0^n$ (note that the subscript convention for $\boldsymbol{\xi}$ corresponds to that used for $\mathbf{x}$). Therefore $\boldsymbol{\xi}_0^n$ is not at the same level of approximation as $\mathbf{x}^{n+1}$ ($= \mathbf{x}_2^{n+1}$) in Equation 45. This is the case because $\boldsymbol{\xi}_0^n$ is the projection of the true solution of Equation 37 at time step n, while $\mathbf{x}_2^{n+1}$ includes both linearization error and state reduction error. For simplicity, we denote the right hand side of Equation 45 as $\mathbf{b}^{n+1}$ (this notation will be used in the subsequent analysis). Then Equation 45 becomes simply $\mathbf{J}^{i+1}\mathbf{x}^{n+1} = \mathbf{b}^{n+1}$.

A.4.3 Constraint Reduction

Applying the POD representation to the left hand side of Equation 45 results in an overdetermined system, which has n_v equations but only l unknowns. This approximation introduces a residual term $\mathbf{r}$, which appears as follows:

$$\mathbf{J}^{i+1}\mathbf{\Phi}\boldsymbol{\xi}^{n+1} = \mathbf{J}^{i+1}\mathbf{\Phi}\boldsymbol{\xi}^{i+1} - \left[\mathbf{A}^{i+1}\mathbf{\Phi}\left(\boldsymbol{\xi}_0^n - \boldsymbol{\xi}^i\right) + \mathbf{B}^{i+1}\left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right)\right] + \mathbf{r}. \tag{46}$$

In general there is no solution for $\boldsymbol{\xi}^{n+1}$ that can render the residual term to be identically zero. Therefore, Equation 46 is usually solved by requiring $\mathbf{r}$ to be zero in an l-dimensional subspace whose basis matrix is denoted $\mathbf{\Psi}^{i+1}$ (i.e., $(\mathbf{\Psi}^{i+1})^T\mathbf{r} = \mathbf{0}$). The matrix $\mathbf{\Psi}^{i+1} \in \mathbb{R}^{n_v \times l}$ is called the constraint reduction matrix, also referred to as the left projection matrix or the test function. Premultiplying Equation 46 by $(\mathbf{\Psi}^{i+1})^T$, with $(\mathbf{\Psi}^{i+1})^T\mathbf{r} = \mathbf{0}$ yields, after some rearrangement

$$\boldsymbol{\xi}^{n+1} = \boldsymbol{\xi}^{i+1} - \left(\mathbf{J}_r^{i+1}\right)^{-1}\left[\mathbf{A}_r^{i+1}\left(\boldsymbol{\xi}_0^n - \boldsymbol{\xi}^i\right) + \mathbf{B}_r^{i+1}\left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right)\right], \tag{47}$$

where the reduced derivative matrices are defined as

$$\mathbf{J}_r^{i+1} = \left(\mathbf{\Psi}^{i+1}\right)^T\mathbf{J}^{i+1}\mathbf{\Phi}, \quad \mathbf{A}_r^{i+1} = \left(\mathbf{\Psi}^{i+1}\right)^T\mathbf{A}^{i+1}\mathbf{\Phi}, \quad \mathbf{B}_r^{i+1} = \left(\mathbf{\Psi}^{i+1}\right)^T\mathbf{B}^{i+1}. \tag{48}$$

Here $\mathbf{J}_r^{i+1} \in \mathbb{R}^{l\times l}$ is the reduced Jacobian matrix, $\mathbf{A}_r^{i+1} \in \mathbb{R}^{l\times l}$, and $\mathbf{B}_r^{i+1} \in \mathbb{R}^{l\times n_u}$.

The choice of $\mathbf{\Psi}^{i+1}$ is not unique. In previous work involving POD-TPWL models for oil-water systems, Galerkin projection was applied [16, 41, 42], meaning we take $\mathbf{\Psi}^{i+1} = \mathbf{\Phi}$. As discussed earlier, this can lead to unstable POD-TPWL models [41]. Recent work has demonstrated that Petrov-Galerkin projection, where $\mathbf{\Psi}^{i+1} = \mathbf{J}^{i+1}\mathbf{\Phi}$, represents a viable (and more numerically stable) alternative [19, 39]. In this work we will investigate the accuracy and stability of these, and other, constraint reduction methods.

Equation 47 defines the POD-TPWL model given the true reduced solution ($\boldsymbol{\xi}_0^n$) at the previous

time step. This equation can also be written as

$$\boldsymbol{\xi}^{n+1} = \left(\mathbf{J}_r^{i+1}\right)^{-1} \left(\boldsymbol{\Psi}^{i+1}\right)^T \mathbf{b}^{n+1}, \tag{49}$$

with $\mathbf{b}^{n+1}$ as defined above. We refer to $\boldsymbol{\xi}^{n+1}$ in Equations 47 and 49 as $\boldsymbol{\xi}_3^{n+1}$. The full-order solution at this level of approximation, denoted $\mathbf{x}_3^{n+1}$, can be reconstructed as $\mathbf{x}_3^{n+1} = \boldsymbol{\Phi}\boldsymbol{\xi}_3^{n+1}$. The error incurred in this step is defined as $\mathbf{e}_{23}^{n+1} = \mathbf{x}_2^{n+1} - \mathbf{x}_3^{n+1}$, and is referred to as the constraint reduction error.

A.4.4 Error Propagation

The term $\boldsymbol{\xi}_0^n$ in Equation 47 is the projection of the true solution $\mathbf{x}_0^n$, which is generally not available. In an actual POD-TPWL computation, $\boldsymbol{\xi}_0^n$ is replaced by $\boldsymbol{\xi}^n$, which corresponds to the POD-TPWL solution at time step n. We thus write:

$$\boldsymbol{\xi}^{n+1} = \boldsymbol{\xi}^{i+1} - \left(\mathbf{J}_r^{i+1}\right)^{-1} \left[\mathbf{A}_r^{i+1}\left(\boldsymbol{\xi}^n - \boldsymbol{\xi}^i\right) + \mathbf{B}_r^{i+1}\left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right)\right]. \tag{50}$$

Equation 50 defines the POD-TPWL model we actually solve. By combining linearization and order reduction, the original high-order nonlinear system of equations has been transformed to a linear expression in l variables (l is typically ∼100–1000), which can be evaluated in seconds [16, 42].

To apply Equation 50 we must first determine the saved state and control point $(\boldsymbol{\xi}^{i+1}, \boldsymbol{\xi}^i, \mathbf{u}^{i+1})$ around which to linearize. This is usually accomplished by minimizing a measure of distance, denoted $d^{n,j}$, between the test solution $\boldsymbol{\xi}$ at time step n and any saved point $\boldsymbol{\xi}^j$ in the training simulation, that is, $i = \arg\min_j \left(d^{n,j}\right)$. Distance definitions are typically problem specific. Here we use the definition given in [39], which entails a weighted combination of the relative difference in dimensionless time (pore volume injected in this context) and in the reduced water saturation or total mole fraction states ($\boldsymbol{\xi}_S$ or $\boldsymbol{\xi}_z$). See [39] for further details.

In Equation 50, $\boldsymbol{\xi}^n$ corresponds to the POD-TPWL solution at the previous time step, which we designate $\boldsymbol{\xi}_4^n$. The solution to Equation 50 is thus designated $\boldsymbol{\xi}_4^{n+1}$, and its corresponding full-order state is $\mathbf{x}_4^{n+1}$. The difference between $\boldsymbol{\xi}_0^n$ and $\boldsymbol{\xi}_4^n$ can be expressed as

$$\boldsymbol{\xi}_0^n - \boldsymbol{\xi}_4^n = \boldsymbol{\Phi}^T(\mathbf{x}_0^n - \mathbf{x}_4^n) = \boldsymbol{\Phi}^T\mathbf{e}_{04}^n. \tag{51}$$

The fact that $\mathbf{e}_{04}^n$ is nonzero results in a difference between the reduced solutions $\boldsymbol{\xi}_3^{n+1}$ and $\boldsymbol{\xi}_4^{n+1}$, and thus a difference between the full-order solutions $\mathbf{x}_3^{n+1}$ and $\mathbf{x}_4^{n+1}$. The error incurred at this step is defined as $\mathbf{e}_{34}^{n+1} = \mathbf{x}_3^{n+1} - \mathbf{x}_4^{n+1}$, which can be viewed as the error inherited from the previous time step. It is given by

$$\begin{aligned}
\mathbf{e}_{34}^{n+1} &= \mathbf{x}_3^{n+1} - \mathbf{x}_4^{n+1} \\
&= \boldsymbol{\Phi}\left(\boldsymbol{\xi}_3^{n+1} - \boldsymbol{\xi}_4^{n+1}\right) \\
&= -\boldsymbol{\Phi}\left(\mathbf{J}_r^{i+1}\right)^{-1} \mathbf{A}_r^{i+1}\left(\boldsymbol{\xi}_0^n - \boldsymbol{\xi}_4^n\right) \\
&= \mathbf{M}^{i+1}\mathbf{e}_{04}^n,
\end{aligned} \tag{52}$$

where $\mathbf{M}^{i+1} = -\mathbf{\Phi}\left(\mathbf{J}_r^{i+1}\right)^{-1}\mathbf{A}_r^{i+1}\mathbf{\Phi}^T$. The third step in Equation 52 corresponds to subtracting Equation 50 (for $\boldsymbol{\xi}_4^{n+1}$) from Equation 47 (for $\boldsymbol{\xi}_3^{n+1}$).

A.4.5 Total Error of POD-TPWL Model

The total error $\mathbf{e}_{04}^{n+1}$ of the POD-TPWL solution $\mathbf{x}_4^{n+1}$ (reconstructed from the solution of Equation 50, $\boldsymbol{\xi}_4^{n+1}$) relative to the true solution of Equation 37, $\mathbf{x}_0^{n+1}$, can be expressed as

$$\begin{aligned}\mathbf{e}_{04}^{n+1} &= \mathbf{x}_0^{n+1} - \mathbf{x}_4^{n+1} \\ &= (\mathbf{x}_0^{n+1} - \mathbf{x}_1^{n+1}) + (\mathbf{x}_1^{n+1} - \mathbf{x}_2^{n+1}) + (\mathbf{x}_2^{n+1} - \mathbf{x}_3^{n+1}) + (\mathbf{x}_3^{n+1} - \mathbf{x}_4^{n+1}) \\ &= \mathbf{e}_{01}^{n+1} + \mathbf{e}_{12}^{n+1} + \mathbf{e}_{23}^{n+1} + \mathbf{e}_{34}^{n+1} \\ &= \mathbf{e}_{01}^{n+1} + \mathbf{e}_{12}^{n+1} + \mathbf{e}_{23}^{n+1} + \mathbf{M}^{i+1}\mathbf{e}_{04}^{n}.\end{aligned} \tag{53}$$

In other words, the total error $\mathbf{e}_{04}^{n+1}$ is the sum of the linearization error $\mathbf{e}_{01}^{n+1}$, the state reduction error $\mathbf{e}_{12}^{n+1}$, the constraint reduction error $\mathbf{e}_{23}^{n+1}$, and the error propagated from the previous time step, $\mathbf{M}^{i+1}\mathbf{e}_{04}^{n}$. The norm of the total error is bounded by the sum of the norms of these four error components, that is

$$\left\|\mathbf{e}_{04}^{n+1}\right\| \leq \left\|\mathbf{e}_{01}^{n+1}\right\| + \left\|\mathbf{e}_{12}^{n+1}\right\| + \left\|\mathbf{e}_{23}^{n+1}\right\| + \left\|\mathbf{M}^{i+1}\mathbf{e}_{04}^{n}\right\|. \tag{54}$$

The linearization error $\mathbf{e}_{01}^{n+1}$ is related to the nonlinearity of the problem and the distance between the current solution and the point in the training run used for linearization. This error component can be reduced by using additional training simulations for linearization (or occasional retraining), by using a better point selection scheme, or by modeling higher-order terms [24]. The second-order terms (SOTs) appearing in the Taylor-series expansion of Equation 37 involve sparse third-order tensors multiplied by two vectors of differences. For example, the second-order term with respect to ($\mathbf{x}^{i+1}$, $\mathbf{u}^{i+1}$) can be written as

$$\text{SOT} = \left(\mathbf{x}^{n+1} - \mathbf{x}^{i+1}\right)^T \frac{\partial^2 \mathbf{g}^{i+1}}{\partial \mathbf{x}^{i+1}\partial \mathbf{u}^{i+1}}\left(\mathbf{u}^{n+1} - \mathbf{u}^{i+1}\right). \tag{55}$$

These second-order derivatives are not usually available in subsurface flow simulators, though it should be possible to construct them using simulators based on automatic differentiation, such as Stanford's Automatic Differentiation-based General Purpose Research Simulator (AD-GPRS) [79]. In addition, for specific applications, some of the second-order terms vanish. For example, for reservoir simulation problems, the cross terms involving second derivatives with respect to ($\mathbf{x}^i$, $\mathbf{x}^{i+1}$) and ($\mathbf{x}^i$, $\mathbf{u}^{i+1}$) are usually zero. Furthermore, for models with standard well treatments in which well pressures (BHPs) are the control parameters, the second-order derivative with respect to $\mathbf{u}^{i+1}$ is zero. Thus only some of the SOTs would need to be considered.

The state reduction error $\mathbf{e}_{12}^{n+1}$ results from applying order reduction to the states. Equations 45 and 41 are linear systems with the same coefficient matrix but different right hand sides. Therefore,

$\left\|\mathbf{e}_{12}^{n+1}\right\| \leq \kappa \left\|\mathbf{x}_2^{n+1}\right\| \|\Delta\mathbf{b}\| / \left\|\mathbf{b}^{n+1}\right\|$, where κ is the condition number of $\mathbf{J}^{i+1}$, $\mathbf{b}^{n+1}$ is the right hand side of Equation 45, and $\Delta\mathbf{b}$ is the difference between the right hand sides of Equations 45 and 41 [25]. The magnitude of $\|\Delta\mathbf{b}\|$ is determined by the state reduction basis matrix $\mathbf{\Phi}$. Therefore, within the framework of POD, the state reduction error $\mathbf{e}_{12}^{n+1}$ can be decreased by improving the quality of $\mathbf{\Phi}$. This can be accomplished, for example, by collecting more snapshots from additional training simulations. Another approach, proposed in [41], is to keep variables in important grid blocks (e.g., well blocks) in the full-order space, meaning these variables are not represented via $\mathbf{x} = \mathbf{\Phi}\boldsymbol{\xi}$. This treatment was shown to provide more accurate results in many cases.

The error terms $\mathbf{e}_{01}^{n+1}$ and $\mathbf{e}_{12}^{n+1}$ are incurred prior to the introduction of constraint reduction in Equation 47, so they do not depend on the matrix $\mathbf{\Psi}^{i+1}$. The error $\mathbf{e}_{23}^{n+1}$ is however incurred by the state and constraint reduction at time step $n+1$, so it does depend on $\mathbf{\Psi}^{i+1}$. The error $\mathbf{e}_{34}^{n+1} = \mathbf{M}^{i+1}\mathbf{e}_{04}^{n}$ is the error propagated from the previous time step. It also depends on $\mathbf{\Psi}^{i+1}$ via the matrix $\mathbf{M}^{i+1}$. An optimal choice of $\mathbf{\Psi}^{i+1}$ should minimize the sum of $\mathbf{e}_{23}^{n+1}$ and $\mathbf{e}_{34}^{n+1}$. The optimal $\mathbf{\Psi}^{i+1}$ will however depend on the error at the previous time step ($\mathbf{e}_{04}^{n}$), which is, in general, not available. Therefore our strategy here is to choose a constraint reduction matrix $\mathbf{\Psi}^{i+1}$ that minimizes the one-step error term $\mathbf{e}_{23}^{n+1}$, while also ensuring that the resulting POD-TPWL model behaves stably, which means the error from previous time steps will not grow unphysically with time.

A.5 Optimal Constraint Reduction Procedures

We now derive optimal constraint reduction matrices that minimize the one-step error $\mathbf{e}_{23}^{n+1}$ in different norms. In our development here, since we only consider one time step, we drop the superscripts $i+1$ and $n+1$.

A.5.1 General Development

The optimality condition can be written as

$$\mathbf{\Psi}^* = \arg\min_{\mathbf{\Psi}} \|\mathbf{e}_{23}\|_{\mathbf{\Theta}}^2 = \arg\min_{\mathbf{\Psi}} \|\mathbf{x}_2 - \mathbf{x}_3\|_{\mathbf{\Theta}}^2 , \tag{56}$$

where $\mathbf{\Psi}^*$ designates the optimum (we drop the superscript $*$ in subsequent equations). Here $\|\cdot\|_{\mathbf{\Theta}}$ is a norm defined as $\|\mathbf{e}\|_{\mathbf{\Theta}} = \sqrt{\mathbf{e}^T\mathbf{\Theta}\mathbf{e}}$, with $\mathbf{e} \in \mathbb{R}^{n_v \times 1}$ and $\mathbf{\Theta} \in \mathbb{R}^{n_v \times n_v}$, where $\mathbf{\Theta}$ is a symmetric positive definite (SPD) matrix and n_v is the dimension of the full-order state $\mathbf{x}$. The requirement for $\mathbf{\Theta}$ to be SPD ensures that the minimum of $\|\mathbf{e}\|_{\mathbf{\Theta}}^2$ is uniquely $\mathbf{e} = \mathbf{0}$.

We denote the objective function of the minimization problem in Equation 56 as $f(\mathbf{\Psi})$, that is, $f(\mathbf{\Psi}) = \|\mathbf{e}_{23}\|_{\mathbf{\Theta}}^2 = \|\mathbf{x}_2 - \mathbf{x}_3\|_{\mathbf{\Theta}}^2$. From Equation 45 we have $\mathbf{x}_2 = \mathbf{J}^{-1}\mathbf{b}$, while $\mathbf{x}_3$ can be reconstructed from the solution of Equation 47 as $\mathbf{x}_3 = \mathbf{\Phi}\boldsymbol{\xi}_3$ (which we denote simply as $\mathbf{\Phi}\boldsymbol{\xi}$ in this

section). Therefore, $f(\boldsymbol{\Psi})$ can be expressed as

$$\begin{aligned} f(\boldsymbol{\Psi}) &= \|\mathbf{x}_2 - \mathbf{x}_3\|_{\boldsymbol{\Theta}}^2 \\ &= \left\|\mathbf{J}^{-1}\mathbf{b} - \boldsymbol{\Phi}\boldsymbol{\xi}\right\|_{\boldsymbol{\Theta}}^2 \\ &= (\mathbf{J}^{-1}\mathbf{b} - \boldsymbol{\Phi}\boldsymbol{\xi})^T\boldsymbol{\Theta}(\mathbf{J}^{-1}\mathbf{b} - \boldsymbol{\Phi}\boldsymbol{\xi}). \end{aligned} \tag{57}$$

The first-order optimality condition for the minimization problem in Equation 56 is $\frac{\partial f}{\partial \boldsymbol{\Psi}} = \mathbf{0}$. From the chain rule and the fact that $\boldsymbol{\xi} = \boldsymbol{\xi}(\boldsymbol{\Psi})$ (since $\boldsymbol{\xi}$ depends on $\boldsymbol{\Psi}$ via $\mathbf{J}_r$, $\mathbf{A}_r$ and $\mathbf{B}_r$), we have $\frac{\partial f}{\partial \boldsymbol{\Psi}} = \frac{\partial f}{\partial \boldsymbol{\xi}}\frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{\Psi}}$, in which case $\frac{\partial f}{\partial \boldsymbol{\xi}} = \mathbf{0}$ is sufficient to conclude that $\frac{\partial f}{\partial \boldsymbol{\Psi}} = \mathbf{0}$. The condition $\frac{\partial f}{\partial \boldsymbol{\xi}} = \mathbf{0}$ gives

$$\boldsymbol{\Phi}^T\boldsymbol{\Theta}\boldsymbol{\Phi}\boldsymbol{\xi} - \boldsymbol{\Phi}^T\boldsymbol{\Theta}\mathbf{J}^{-1}\mathbf{b} = \mathbf{0}, \tag{58}$$

or equivalently,

$$\boldsymbol{\xi} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Theta}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Theta}\mathbf{J}^{-1}\mathbf{b}. \tag{59}$$

We also have a direct expression for $\boldsymbol{\xi}$ from Equation 49:

$$\boldsymbol{\xi} = \left(\boldsymbol{\Psi}^T\mathbf{J}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Psi}^T\mathbf{b}. \tag{60}$$

Equations 60 and 59 must both hold for an arbitrary vector $\mathbf{b}$. Therefore the following matrix equation for $\boldsymbol{\Psi}$ applies

$$\left(\boldsymbol{\Psi}^T\mathbf{J}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Psi}^T = \left(\boldsymbol{\Phi}^T\boldsymbol{\Theta}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Theta}\mathbf{J}^{-1}. \tag{61}$$

The solution to Equation 61 is, by inspection,

$$\boldsymbol{\Psi}^T = \boldsymbol{\Phi}^T\boldsymbol{\Theta}\mathbf{J}^{-1}. \tag{62}$$

Note that $\boldsymbol{\Psi}^T = \mathbf{C}\boldsymbol{\Phi}^T\boldsymbol{\Theta}\mathbf{J}^{-1}$, for any full-rank $\mathbf{C} \in \mathbb{R}^{l\times l}$, is also a solution to Equation 61. The choice of $\mathbf{C}$ will not affect the solution for $\boldsymbol{\xi}$ since $\mathbf{C}^{-1}\mathbf{C}$ immediately appears in Equation 60 (here we take $\mathbf{C} = \mathbf{I}$).

Equation 62 provides a general solution for $\boldsymbol{\Psi}^T$ that satisfies the optimality condition in Equation 56. The solution for $\boldsymbol{\Psi}^T$ in Equation 62 depends on the matrix $\boldsymbol{\Theta}$ and in general involves the matrix $\mathbf{J}^{-1}$. In practice, special choices of $\boldsymbol{\Theta}$ which eliminate $\mathbf{J}^{-1}$ are often employed. Examples are Galerkin projection and Petrov-Galerkin projection, which we will describe in detail below. Other choices for $\boldsymbol{\Theta}$, which do not eliminate $\mathbf{J}^{-1}$, will lead to higher computational cost, but they may have better theoretical properties and display better numerical performance. We will discuss two such treatments, inverse projection and weighted inverse projection, in Section A.8.

A.5.2 Galerkin Projection

Galerkin projection corresponds to the case where $\boldsymbol{\Theta} = \mathbf{J}$. Equation 62 then becomes $\boldsymbol{\Psi}^T = \boldsymbol{\Phi}^T$. Galerkin projection has been used in many previous POD-based order reduction procedures [12,

15, 16, 22, 41, 48, 60, 67].

Galerkin projection enforces so-called Galerkin orthogonality, which means that the residual vector of Equation 45 is orthogonal to the state subspace $\mathbf{\Phi}$. It also minimizes the objective function $(\mathbf{e}_{23})^T \mathbf{J}\mathbf{e}_{23}$. When the matrix $\mathbf{J}$ is SPD, this projection method is optimal in the sense defined in Equation 56. However, if $\mathbf{J}$ is not SPD, $(\mathbf{e}_{23})^T \mathbf{J}\mathbf{e}_{23}$ will not be a norm definition for $\mathbf{e}_{23}$ since it could be negative. In subsurface flow simulations involving multiple phases, the Jacobian matrix is in general not SPD. Therefore Galerkin projection is in general not a strict minimizer of the norm of $\mathbf{e}_{23}$.

However, despite this theoretical limitation, previous applications of POD for subsurface flow [12, 15, 16, 22, 41, 48, 60, 67] using Galerkin projection often show reasonable accuracy, especially in terms of the well production and injection rates. This may be due to the fact that the Galerkin orthogonality condition requires the residual vector to be orthogonal to the columns of $\mathbf{\Phi}$. Recall that the columns of $\mathbf{\Phi}$ capture the variations in the states since they are computed through application of POD. Therefore, variables with larger variation tend to have larger weights in the basis vectors. As a result, the corresponding equations are weighted more heavily and are thus solved more accurately. In reservoir simulation applications, variables with large variation often correspond to well blocks and to blocks in the near-well regions. The additional weighting applied to these blocks may enable Galerkin projection to provide accurate well rate predictions.

A.5.3 Petrov-Galerkin Projection

If $\mathbf{\Theta}$ is taken as $\mathbf{J}^T\mathbf{J}$, Equation 62 becomes $\mathbf{\Psi}^T = \mathbf{\Phi}^T\mathbf{J}^T$. This projection method, called Petrov-Galerkin projection, has been used in [10] for order reduction of large-scale systems with high-dimensional parametric input and in [19] for order reduction within the context of DEIM. Petrov-Galerkin projection has some interesting properties. First, because $\mathbf{J}^T\mathbf{J}$ is SPD, the resulting $\mathbf{\Psi}$ satisfies Equation 56 and minimizes the norm of $\mathbf{e}_{23}$, with the norm defined as $\|\mathbf{e}\|^2_{\mathbf{J}^T\mathbf{J}} = \mathbf{e}^T\mathbf{J}^T\mathbf{J}\mathbf{e}$. Second, Petrov-Galerkin projection is equivalent to solving the normal equation for the overdetermined system in Equation 46, which minimizes the 2-norm of the residual vector.

The optimality condition for Petrov-Galerkin projection can also be interpreted in another way. With the Petrov-Galerkin method, Equation 56 is equivalent to

$$\mathbf{\Psi}^* = \arg\min_{\mathbf{\Psi}} \|\mathbf{e}_{23}\|^2_{\mathbf{J}^T\mathbf{J}} = \arg\min_{\mathbf{\Psi}} \|\mathbf{J}\mathbf{e}_{23}\|_2^2 . \tag{63}$$

In other words, this approach minimizes the 2-norm of $\mathbf{J}\mathbf{e}_{23}$. Writing the singular value decomposition of $\mathbf{J}$ as $\mathbf{J} = \mathbf{U}_J\mathbf{\Sigma}_J\mathbf{V}_J^T$, where $\mathbf{U}_J$ is an orthogonal matrix containing as its columns the left-singular vectors of $\mathbf{J}$, $\mathbf{\Sigma}_J$ is a diagonal matrix with the singular values of $\mathbf{J}$ on the diagonal, and $\mathbf{V}_J$ is an orthogonal matrix containing as its columns the right-singular vectors of $\mathbf{J}$ (or equivalently, the eigenvectors of $\mathbf{J}^T\mathbf{J}$), Equation 63 can be expressed as

$$\mathbf{\Psi}^* = \arg\min_{\mathbf{\Psi}} \left\|\mathbf{\Sigma}_J\mathbf{V}_J^T\mathbf{e}_{23}\right\|_2^2 . \tag{64}$$

Equation 64 indicates that the Petrov-Galerkin method transforms the error vector into a coordinate defined by $\mathbf{V}_J$, weights each element of the new vector with the corresponding singular value, and then minimizes the 2-norm of the weighted transformed error vector. In subsurface flow simulation, the Jacobian matrix $\mathbf{J}$ is usually very ill-conditioned and its singular values vary widely, spanning up to 10 orders of magnitude. Therefore the weighting scheme using singular values can be highly skewed. In other words, some components in the error vector will be very strongly weighted while others may be very weakly weighted. Therefore, although the Petrov-Galerkin method is optimal in the sense defined in Equation 56, the skewed weighting embedded in the method may result in inaccuracy in the resulting POD-TPWL model for some quantities.

A.6 Stability Criteria

In addition to accuracy and efficiency considerations, the constraint reduction matrix $\mathbf{\Psi}$ should also be selected to ensure the stability of the resulting POD-TPWL method. In theory, we could attempt to construct a SPD matrix $\mathbf{\Theta}$ in Equation 62 that also satisfies the Lyapunov stability criteria. This would provide a guaranteed stable and optimally accurate $\mathbf{\Psi}$. This approach, however, would entail coupling Equation 62 with the Lyapunov equations and solving the resulting matrix equations. Such an approach is unlikely to be computationally tractable for high-dimensional subsurface flow problems. Therefore, in this work we start with constraint reduction methods that satisfy Equation 62, and then consider methods to assess and enhance their stability.

The stability of reduced-order models based on TPWL was first considered in [62], which addressed order reduction of nonlinear ordinary differential equation (ODE) systems. In that work, the piecewise linear reduced-order ODE system was deemed stable when the coefficient matrix of the linear ODE system at each linearization step is a Hurwitz matrix (all eigenvalues have negative real part). Bond and Daniel [5] considered reduced-order models for linear time-invariant ODE systems and proposed that $\mathbf{\Psi}$ be chosen to guarantee stability by satisfying the Lyapunov equations of the reduced system. However, their approach involves solving a matrix optimization problem, which for our models would entail an optimization of $n_v \times l$ variables under constraints for each time step of the training simulation. As indicated above, this amount of computation will be prohibitive for models of reasonable size.

He *et al.* [41] considered the stability of POD-TPWL for fully discretized (PDE) systems in the context of reservoir simulation. There it was shown that, in order for the POD-TPWL model to be stable, the amplification factor at each time step of a particular matrix should be less than 1. This analysis will now be described within the framework of error assessment.

The stability of the POD-TPWL formulation can be analyzed from the error propagation expression (Equation 53), which can be rewritten as

$$\mathbf{e}_{04}^{n+1} = \mathbf{M}^{i+1}\mathbf{e}_{04}^{n} + \mathbf{e}_{03}^{n+1}, \tag{65}$$

where

$$\mathbf{M}^{i+1} = -\mathbf{\Phi}\left(\mathbf{J}_r^{i+1}\right)^{-1}\mathbf{A}_r^{i+1}\mathbf{\Phi}^T, \tag{66}$$

with $\mathbf{M}^{i+1} \in \mathbb{R}^{n_v \times n_v}$ (note that $\mathbf{M}^{i+1}$ appeared originally in Equation 52). Equation 65 indicates that the total error of the POD-TPWL model at time step $n+1$ is the total error at time step n amplified by the matrix $\mathbf{M}^{i+1}$, which is referred to as the amplification matrix, plus the one-step error $\mathbf{e}_{03}^{n+1}$ ($\mathbf{e}_{03}^{n+1} = \mathbf{e}_{01}^{n+1} + \mathbf{e}_{12}^{n+1} + \mathbf{e}_{23}^{n+1}$) incurred at time step $n+1$. It is important to observe that $\mathbf{M}^{i+1}$ in Equation 66 is defined as a projection of the matrix product $-\left(\mathbf{J}_r^{i+1}\right)^{-1}\mathbf{A}_r^{i+1}$, whose dimension is $l \times l$. Therefore, although $\mathbf{M}^{i+1} \in \mathbb{R}^{n_v \times n_v}$, its rank is at most l.

We denote the spectral radius of $\mathbf{M}^{i+1}$ as γ^{i+1} and refer to it as the amplification factor. For a matrix $\mathbf{M}$ that does not vary with time (i.e., $\mathbf{M}^{i+1} = \mathbf{M}$), the error $\mathbf{e}_{04}^{n}$ in Equation 65 will not grow exponentially if and only if the amplification factor γ^{i+1} is less than or equal to 1 [34]. In a piecewise linear system $\mathbf{M}^{i+1}$ generally changes at each time step. However, the amplification factor γ^{i+1} was still shown to be a strong indicator of stability. Specifically, it was demonstrated in [41] that an isolated γ^{i+1} that is greater than 1 may amplify the error at a specific time step and create a spike in the solution. Several consecutive time steps with $\gamma^{i+1} > 1$ may cause the solution to become unphysical. In practice, however, the occasional occurrence of γ^{i+1} values that only slightly exceed 1 does not appear to cause numerical instability. Therefore, to ensure that the error does not amplify over time, we require that γ^{i+1} not exceed 1 by more than a small threshold; e.g., $\gamma^{i+1} < 1.05$. This will be used as the criterion to assure (essentially) stable POD-TPWL behavior in this work. We note that [62] and [6] also used the spectral radius of an amplification matrix as a stability indicator for piecewise linear reduced-order ODE systems.

Since $\mathbf{M}^{i+1}$ is a high-dimensional matrix, its spectral radius can be expensive to compute. Fortunately, because of the rank-deficiency of $\mathbf{M}^{i+1}$ ($\mathbf{M}^{i+1}$ has a maximum rank of l), we do not need to analyze this high-dimensional matrix to assess POD-TPWL stability. Rather, we define $\mathbf{M}_r^{i+1} \in \mathbb{R}^{l \times l}$ as

$$\mathbf{M}_r^{i+1} = -\left(\mathbf{J}_r^{i+1}\right)^{-1}\mathbf{A}_r^{i+1}. \tag{67}$$

We now show that $\mathbf{M}^{i+1}$ and $\mathbf{M}_r^{i+1}$ have the same nonzero eigenvalues, and thus the same spectral radius.

From the definitions of $\mathbf{M}^{i+1}$ and $\mathbf{M}_r^{i+1}$ in Equations 66 and 67, we see that $\mathbf{M}^{i+1} = \mathbf{\Phi}\mathbf{M}_r^{i+1}\mathbf{\Phi}^T$. In addition, using the fact that $\mathbf{\Phi}$ is orthonormal (i.e., $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{I}$), we have

$$\mathbf{M}_r^{i+1} = (\mathbf{\Phi}^T\mathbf{\Phi})\mathbf{M}_r^{i+1}(\mathbf{\Phi}^T\mathbf{\Phi}) = \mathbf{\Phi}^T(\mathbf{\Phi}\mathbf{M}_r^{i+1}\mathbf{\Phi}^T)\mathbf{\Phi} = \mathbf{\Phi}^T\mathbf{M}^{i+1}\mathbf{\Phi}. \tag{68}$$

Using the expressions above, the fact that the matrices $\mathbf{M}^{i+1}$ and $\mathbf{M}_r^{i+1}$ have the same nonzero eigenvalues and thus the same spectral radius can be shown as follows. Let $\mathbf{y}$ be an eigenvector of $\mathbf{M}^{i+1}$ with nonzero eigenvalue λ, that is, $\mathbf{M}^{i+1}\mathbf{y} = \lambda\mathbf{y}$. Using this, along with the relationship between $\mathbf{M}^{i+1}$ and $\mathbf{M}_r^{i+1}$, and premultiplying by $\mathbf{\Phi}^T$, we have

$$\mathbf{\Phi}^T\mathbf{M}^{i+1}\mathbf{y} = \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{M}_r^{i+1}\mathbf{\Phi}^T\mathbf{y} = \mathbf{M}_r^{i+1}\mathbf{\Phi}^T\mathbf{y} = \lambda\mathbf{\Phi}^T\mathbf{y}. \tag{69}$$

This means $\mathbf{\Phi}^T\mathbf{y}$ is an eigenvector of $\mathbf{M}_r^{i+1}$ with eigenvalue λ. Similarly, if $\boldsymbol{\eta}$ is an eigenvector of $\mathbf{M}_r^{i+1}$ with nonzero eigenvalue λ, we have

$$\mathbf{M}^{i+1}\mathbf{\Phi}\boldsymbol{\eta} = \mathbf{\Phi}\mathbf{M}_r^{i+1}\mathbf{\Phi}^T\mathbf{\Phi}\boldsymbol{\eta} = \mathbf{\Phi}\mathbf{M}_r^{i+1}\boldsymbol{\eta} = \lambda\mathbf{\Phi}\boldsymbol{\eta}, \tag{70}$$

which means $\mathbf{\Phi}\boldsymbol{\eta}$ is an eigenvector for $\mathbf{M}^{i+1}$ with eigenvalue λ. Therefore, $\mathbf{M}^{i+1}$ and $\mathbf{M}_r^{i+1}$ have the same nonzero eigenvalues and thus the same spectral radius γ^{i+1}. In practice, γ^{i+1} can thus be calculated efficiently by computing the largest eigenvalue of the low-dimensional matrix $\mathbf{M}_r^{i+1}$, as described in [41].

Given the derivative matrices $\mathbf{J}^{i+1}$ and $\mathbf{A}^{i+1}$ and simulation results for all training simulations, the amplification matrix only depends on the number of reduced variables (l_p and l_S for oil-water problems, and l_p and l_z for compositional problems), which determine $\mathbf{\Phi}$ and the constraint reduction matrix $\mathbf{\Psi}$. For general choices of l_p and l_S (or l_z) and $\mathbf{\Psi}$, the resulting POD-TPWL model can be unstable. It was shown in [41] that for oil-water problems the stability behavior of the POD-TPWL model using Galerkin projection can be very sensitive to the choice of l_p and l_S. Improved results for oil-water problems were achieved in [41] by finding the (l_p, l_S) combination that provides the best stability properties. This approach entails specifying minimum and maximum values for l_p and l_S, calculating the maximum amplification factor for all time steps i in the training run ($\max_i \gamma^i$) for all (l_p, l_S) combinations considered, and selecting the (l_p, l_S) combination that provides the lowest $\max_i \gamma^i$. This can be accomplished efficiently (in low-dimensional space) because the reduced derivative matrices $\mathbf{J}_r^{i+1}$ and $\mathbf{A}_r^{i+1}$ for different (l_p, l_S) combinations are just submatrices of the reduced derivative matrices for the largest values of l_p and l_S considered. Note that this (l_p, l_S) selection method will only be effective when $\max_i \gamma^i$ is below the specified maximum (e.g., 1.05).

This procedure provides an offline indication of POD-TPWL solution stability using only low-order computations. This stability indicator can be applied for any constraint reduction procedure, and we will use it for all of the approaches considered in this Appendix. Along these lines, the stability of POD-TPWL using Petrov-Galerkin projection has not, to our knowledge, been previously studied. In Section A.7 we will see that Petrov-Galerkin projection provides much better stability than Galerkin projection for challenging cases. In addition, in Section A.8, we will show that the two new projection methods introduced here, inverse projection and weighted inverse projection, also provide POD-TPWL models that behave stably.

A.7 Numerical Implementation and Results

We now describe some key aspects of the POD-TPWL implementation and the way in which we quantify error. Numerical results are then presented for two oil-water models and one compositional model. Additional results and discussion can be found in [38].

A.7.1 POD-TPWL Implementation

The POD-TPWL method has been implemented for compatibility with Stanford's Automatic Differentiation-based General Purpose Research Simulator (AD-GPRS) [79]. AD-GPRS was modified to output the state and derivative information required to construct the POD-TPWL model.

During the offline (preprocessing) stage, two or three training simulations are performed using AD-GPRS for specific sets of input (BHP) control parameters. One of the training runs is designated the primary training simulation. This run provides state vectors and derivative matrices for use in subsequent (inline) computations. The other (secondary) training runs are used only to provide snapshots for the construction of the basis matrix $\mathbf{\Phi}$. Secondary training runs are needed because the number of snapshots from a single simulation run, which corresponds to the number of time steps in that run, is typically not enough to provide a high-quality POD basis matrix. More specifically, we have found that around 300 snapshots are needed to construct the POD basis for the problems considered in this Appendix. Given that a typical run entails 100–200 time steps, this corresponds to 2–3 training runs. We do not apply any special procedures to determine the controls used in the primary or secondary training runs (these controls are all generated randomly over the range of interest), though it is possible that better POD-TPWL accuracy could be achieved through use of a more formal approach.

In the basic implementation of the offline procedure, the state vector at each time step of each training simulation, as well as the derivative matrices $\mathbf{J}^i$, $\mathbf{A}^i$ and $\mathbf{B}^i$ at each time step of the primary training simulation, are saved to disk. The POD basis is then constructed from the snapshot matrices. Finally, the reduced states ($\boldsymbol{\xi}^i = \mathbf{\Phi}^T \mathbf{x}^i$) and derivative matrices are formed, with the latter computed using Equation 48.

If full-order derivative matrices are written as output, the storage requirements for the preprocessing computations can be very large. In [39], a reduced-storage offline procedure is described, in which only the reduced-order matrices $\mathbf{J}_r^i$, $\mathbf{A}_r^i$ and $\mathbf{B}_r^i$ are written to disk. This approach requires that the primary training simulation be run twice — once to provide snapshots before construction of the basis matrix $\mathbf{\Phi}$, and once after. The computational cost for the second of these runs can be reduced substantially, however, since the converged states are already known (meaning no iteration or linear solutions are required). See [39] for more details on both the basic and reduced-storage offline procedures.

In the inline stage, Equation 50 is evaluated for control parameters $\mathbf{u}^{n+1}$ that differ from those used in the training runs. After the reduced state $\boldsymbol{\xi}^{n+1}$ is computed, the full-order state $\mathbf{x}^{n+1}$ can be reconstructed at selected locations (e.g., well blocks), and other quantities of interest, such as the phase flow rates for each well, can be calculated. For compositional problems, the calculation of well flow rate additionally requires that a flash calculation be performed at the well blocks to determine secondary variables such as oil saturation. More details can be found in [39].

As indicated above, in addition to performing the training simulations, in the offline stage we must also output detailed information at each time step of the primary training simulation, construct the POD basis, and reduce the training states and derivatives. The reduction of the state

vectors and derivative matrices constitutes the majority of the additional offline overhead. For both the Galerkin and Petrov-Galerkin constraint reduction procedures, this offline cost is approximately equal to the cost of an additional full-order simulation. Once the POD-TPWL model is constructed, however, an inline run typically takes only a few seconds. Thus POD-TPWL is most suitable for use in applications requiring a large number of simulations with different input control parameters, as is the case for production optimization computations.

A.7.2 Error Definitions

In order to assess the accuracy of POD-TPWL models, we compute the mismatch (error) in well flow rates between the full-order AD-GPRS solution (Q_{full}) and the POD-TPWL simulation (Q_{tpwl}). Phase flow-rate errors for a particular well are computed at each time step, then integrated over time, and then normalized by the time-integrated flow rate for that well from the full-order solution. Errors from all wells of the same type (injection or production) are then averaged to provide overall error values. For example, the overall average error for oil production rate, designated E_o, is computed as:

$$E_o = \frac{1}{n_{pw}} \sum_{j=1}^{n_{pw}} \frac{\int_0^T |Q^j_{o,full} - Q^j_{o,tpwl}| \, \mathrm{d}t}{\int_0^T Q^j_{o,full} \, \mathrm{d}t}, \tag{71}$$

where subscript o designates oil, superscript j indicates a particular production well, n_{pw} is the total number of production wells, and T is the total simulation time. The integration is performed using the trapezoidal rule.

For oil-water models, we compute the error in oil production rate E_o, water production rate E_w and water injection rate E_{iw}. For compositional models we calculate the error for oil production rate E_o, gas production rate E_g and gas injection rate E_{ig}. These errors are computed using expressions analogous to Equation 71.

We also calculate the average state error over all time steps and all grid blocks. For example, the average error in pressure E_p is defined as

$$E_p = \frac{1}{n_t n_b} \sum_{i=1}^{n_t} \sum_{k=1}^{n_b} \left| p^i_{k,full} - p^i_{k,tpwl} \right|, \tag{72}$$

where n_t is the number of time steps in the simulation, n_b is the number of grid blocks, $p^i_{k,full}$ is the full-order pressure solution for block k at time step i, and $p^i_{k,tpwl}$ is the analogous quantity for the POD-TPWL model, constructed through application of $\mathbf{x} = \mathbf{\Phi}\boldsymbol{\xi}$. Similar expressions are used for other variables. For oil-water models we compute the average error in pressure and water saturation (E_p and E_S), and for compositional models the average error in pressure and mole fraction of the injected component (E_p and E_z).

A.7.3 Case 1: Oil-Water Flow with Equal Phase Densities

The reservoir model for Cases 1 and 2, shown in Figure 27, is a portion of the so-called Stanford VI reservoir model developed in [20]. The model represents a fluvial depositional system, with high-permeability (sand) channels embedded in a low-permeability background shale. The dimensions of the grid are $30 \times 40 \times 17$, for a total of 20,400 cells. The model contains two injection wells and four production wells, as indicated in Figure 27. The production wells are perforated (open to flow) in the upper five layers and the injection wells in the lower three layers. The wells are controlled through specification of time-varying bottom-hole pressure (BHP).

The relative permeability functions, which quantify the relative amounts of water and oil flow in each grid block, are as follows:

$$k_{rw} = \left(\frac{S_w - S_{wc}}{1 - S_{or} - S_{wc}} \right)^2, \quad k_{ro} = \left(\frac{S_o - S_{or}}{1 - S_{or} - S_{wc}} \right)^2, \tag{73}$$

where oil saturation $S_o = 1 - S_w$. The parameters S_{wc} and S_{or}, specified to be 0.02 and 0.3 respectively, account for the fact that both phases cease to flow below some threshold saturation. The oil and water viscosities (μ_o and μ_w) are 3 cp and 1 cp respectively. These, together with the relative permeability functions, define the phase mobility functions in Equation 36 (specifically, $\lambda_o = k_{ro}/\mu_o$ and $\lambda_w = k_{rw}/\mu_w$). The oil and water phase densities are both specified to be 1000 kg/m^3. The initial reservoir pressure is 5880 psi (405.4 bar) and the initial water saturation is 0.1.

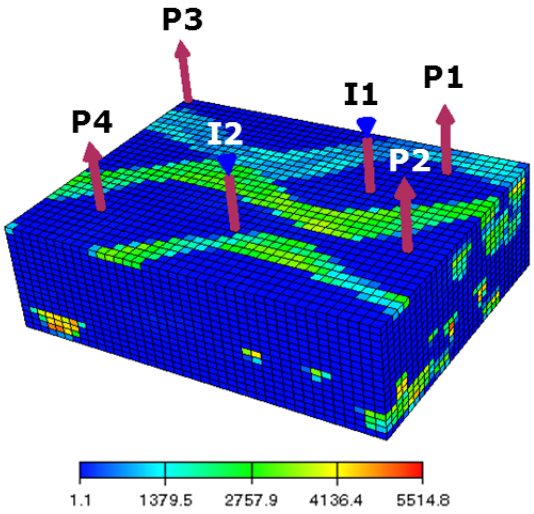
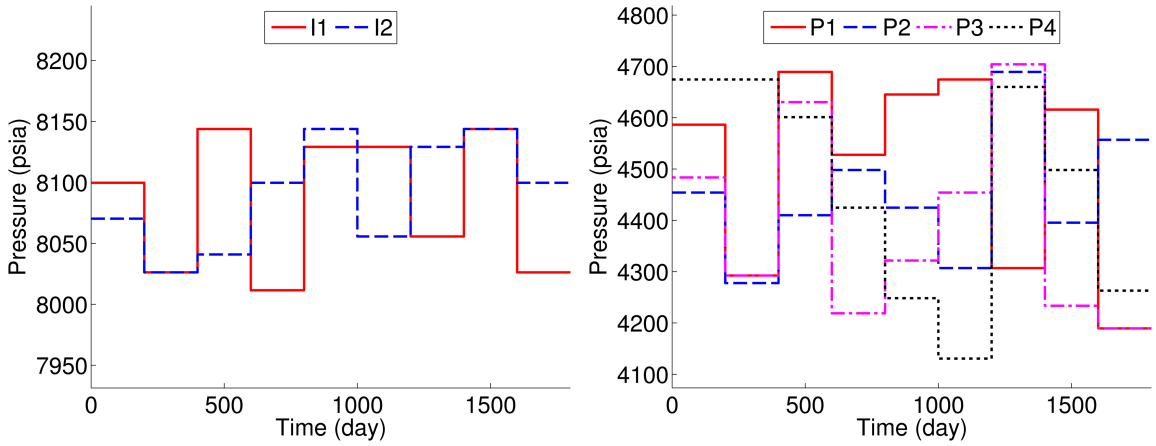


Figure 27: Reservoir model for Cases 1 and 2 (permeability in the x-direction is shown)

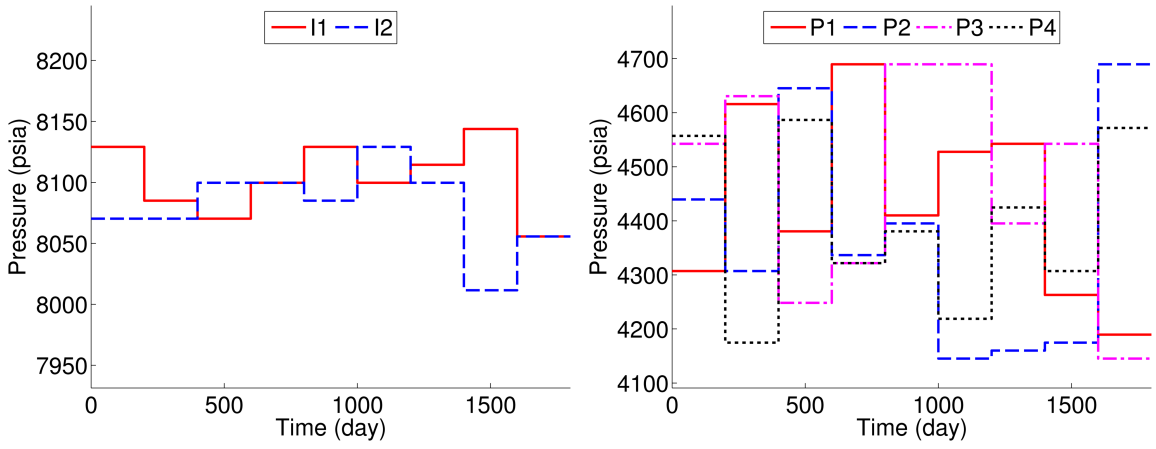
Three training simulations are performed to construct the POD-TPWL model, one of which is used as the primary training run. The BHP controls of the two injectors and four producers are shown in Figure 28. Well settings are varied every 200 days and are generated randomly over a prescribed range. We take this range to be relatively narrow in the examples in this Appendix

in order to focus on constraint reduction error and stability behavior. From the three training simulations, 334 snapshots are collected to construct the basis matrix $\mathbf{\Phi}$. We use 70 reduced pressure variables and 100 reduced water saturation variables ($l_p = 70$, $l_S = 100$). Figure 29 shows the randomly generated BHPs for the test case, which differ from the training-run BHPs. We note that the training and test-case BHP profiles are meant to resemble those generated during a computational optimization procedure, where the goal is to determine the time-varying BHPs that maximize a prescribed measure of reservoir performance.



(a) Injector BHPs (b) Producer BHPs

Figure 28: Time-varying BHPs for the primary training simulation for Cases 1 and 2



(a) Injector BHPs (b) Producer BHPs

Figure 29: Time-varying BHPs for the test simulation for Cases 1 and 2

As discussed in [41], for oil-water systems with equal phase densities, the POD-TPWL model with Galerkin projection tends to behave stably for most (l_p, l_S) combinations. Figure 30 shows the amplification factor γ^i at each time step i for both Galerkin projection (GLK) and Petrov-Galerkin projection (PG). The label 'GLK_70_100' in the legend indicates the result using Galerkin

projection with 70 reduced pressure variables and 100 reduced water saturation variables. The labels for other constraint reduction methods follow this format. It is clear from Figure 30 that the time-varying γ^i for both methods for this case are very close to 1. Indeed, they are below 1 for the entire simulation period, meaning that the resulting POD-TPWL models are always stable. Note that there are small spikes in γ^i every 200 days. These are due to the very small time steps used in the training run when the BHPs are changed. For an infinitely small time step, the Jacobian matrix **J** will equal the negative of the **A** matrix, and the resulting amplification matrix will be the identity matrix, with γ^i of 1. We note that, although we show results only for $l_p = 70$ and $l_S = 100$, similar stable performance can be observed for different l_p and l_S over a reasonable range (as will be illustrated later).

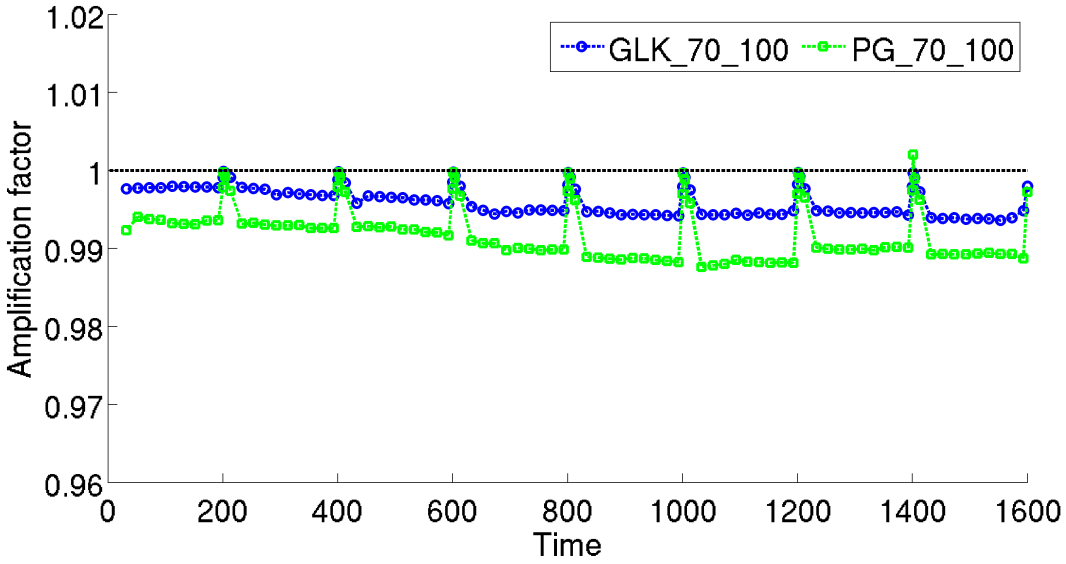


Figure 30: Amplification factor γ^i for each time step in Case 1

We now compare the performance of the Galerkin and Petrov-Galerkin methods for Case 1. Figure 31 shows the oil production rate for Producer 1, and Figure 32 shows the water injection rate for Injector 1. The black dotted line depicts the result from the primary training simulation, about which we linearize. Shown in red is the full-order (reference) AD-GPRS solution for the test case. It is clear that the test and training solutions differ. Galerkin and Petrov-Galerkin projection results are shown in blue (open circles) and green (open squares), respectively. Both approaches provide accurate results in this case, though Galerkin projection can be seen to be slightly more accurate for water injection (see, e.g., results at early time in Figure 32). The results in Figures 31 and 32 are representative of those for the other wells.

Table 2 summarizes the flow rate errors and average state errors from POD-TPWL using the two constraint reduction methods. For the same values of l_p and l_S, Galerkin projection provides better accuracy for this case than Petrov-Galerkin projection in all five error measures. This may be due to inaccuracy resulting from the skewed weighting inherent in Petrov-Galerkin projection,

as discussed in Section A.5.3.

For this example, the full-order parallelized AD-GPRS simulation requires about 290 seconds to run on a cluster node with eight cores (dual quad-core Nehalem™). The POD-TPWL model, with either Galerkin or Petrov-Galerkin projection, requires approximately 0.7 seconds on a single core of the same processor. This corresponds to a runtime speedup of a factor of about 400. As noted earlier, preprocessing requires three training simulations plus overhead computations equal to about an additional full-order simulation.

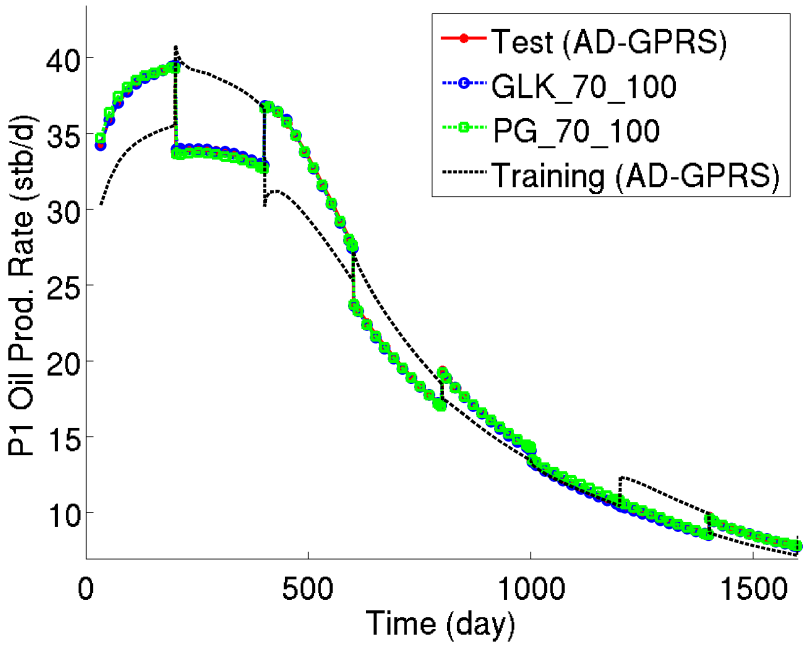


Figure 31: Oil production rate for Producer 1 in Case 1. Results for Test (AD-GPRS), GLK_70_100 and PG_70_100 essentially overlay one another

Table 2: Summary of error for Case 1

	E_o	E_w	E_{iw}	E_p (psi)	E_S
GLK_70_100	0.0054	0.0058	0.0015	1.57	0.00066
PG_70_100	0.0113	0.0156	0.0052	7.07	0.00130

A.7.4 Case 2: Oil-Water Flow with Unequal Phase Densities

In Case 1 both projection methods were stable. However, stability may become an issue when more complicated physics is introduced. This will be illustrated in Case 2, which is identical to Case 1 except that now the oil and water phase densities are set to 800 kg/m^3 and 1000 kg/m^3, respectively. The difference in density results (physically) in gravity-driven countercurrent flow, which leads to changes in the structure of the Jacobian matrix because of upstream weighting [3]. As in Case 1, we perform three training simulations to construct the POD-TPWL model; one

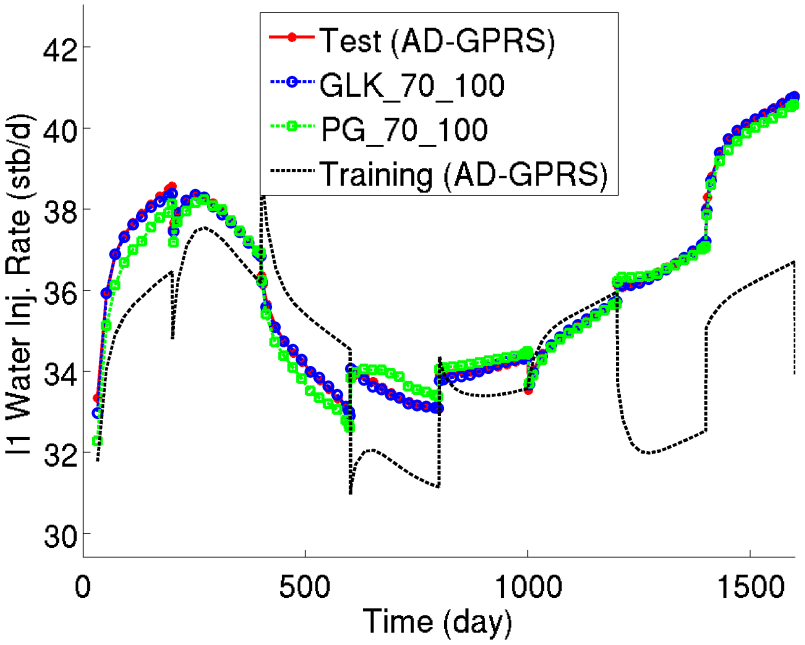


Figure 32: Water injection rate for Injector 1 in Case 1. Results for Test (AD-GPRS) and GLK_70_100 essentially overlay one another

of these is the primary training run. The time-varying BHP controls for the training and test simulations are the same as in Case 1 (see Figures 28 and 29). From the three training simulations, 335 snapshots are collected. We again take $l_p = 70$ and $l_S = 100$.

Figure 33 shows the oil production rate for Producer 1, and Figure 34 shows the water injection rate for Injector 1. It can be seen that the results using Petrov-Galerkin projection match the true test-case solution fairly closely, though some differences are evident. The results using Galerkin projection, by contrast, display substantial fluctuations in the early stage of the simulation. Similar fluctuations for the Galerkin projection run are also observed in flow rates for the other wells (not shown here).

The different performance of the Galerkin and Petrov-Galerkin projections for this case can be explained by Figure 35, which shows the amplification factor γ^i at each time step for the two methods. For the Petrov-Galerkin method, the value of γ^i is below or very near 1 for the entire simulation period. Thus the Petrov-Galerkin method performs stably throughout the simulation. Again, the peaks visible every 200 days are due to the small time steps used when the BHP controls change. For the Galerkin method, the value of γ^i is much larger than 1 at early time, and at some time steps it is as high as 14.5 (note the vertical axis only extends to 1.5). This instability causes the fluctuations evident in Figure 34. At later time the amplification factor decreases to around 1 and the fluctuations disappear.

Note that the stability results in Figure 35 are specifically for $l_p = 70$ and $l_S = 100$. As discussed above, the choice of l_p and l_S affects stability behavior. Figure 36 depicts in log scale the values

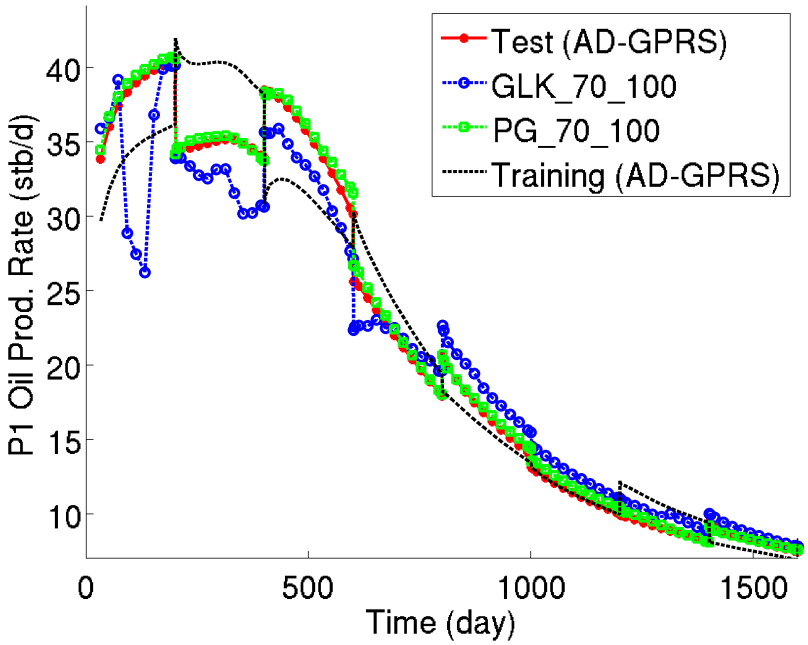


Figure 33: Oil production rate for Producer 1 in Case 2

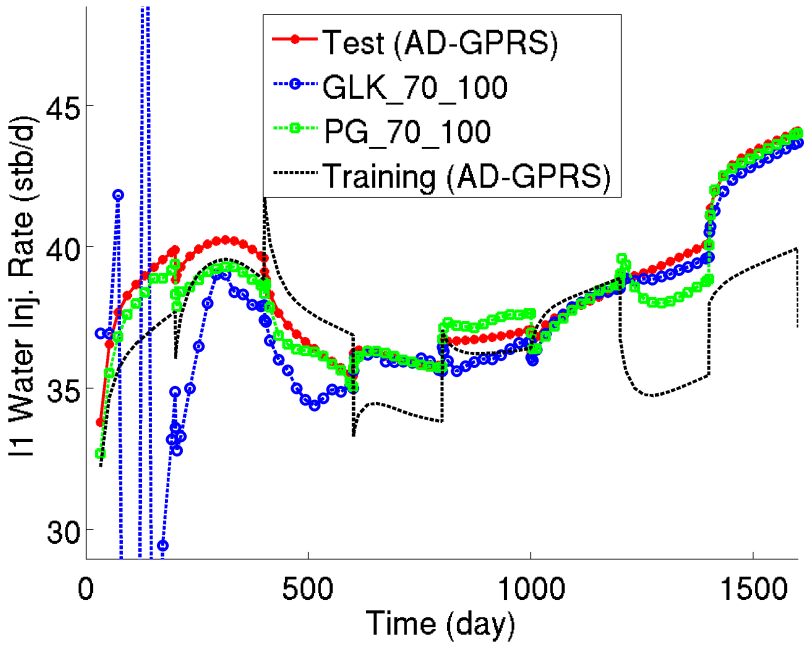


Figure 34: Water injection rate for Injector 1 in Case 2

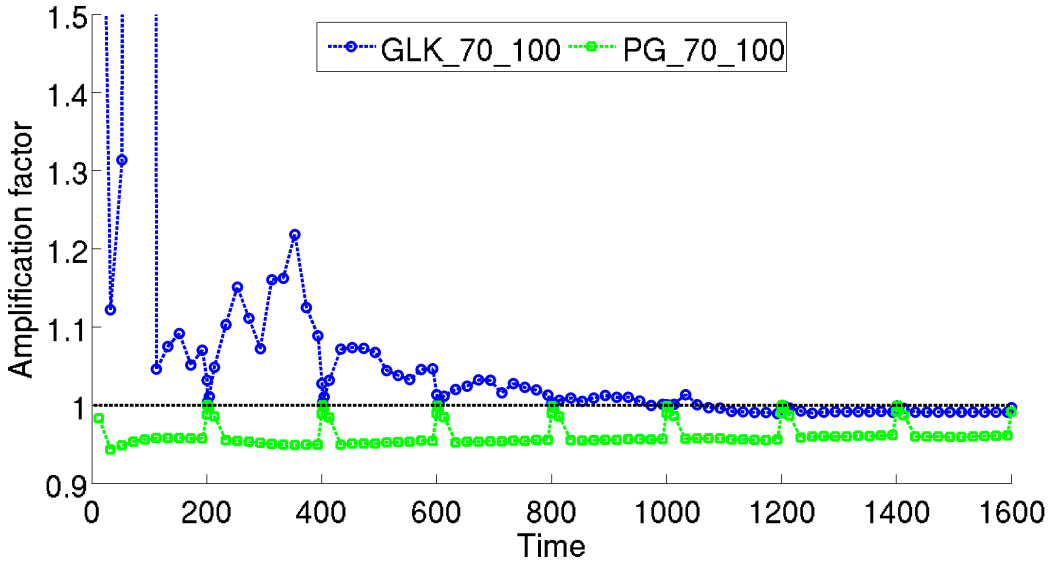


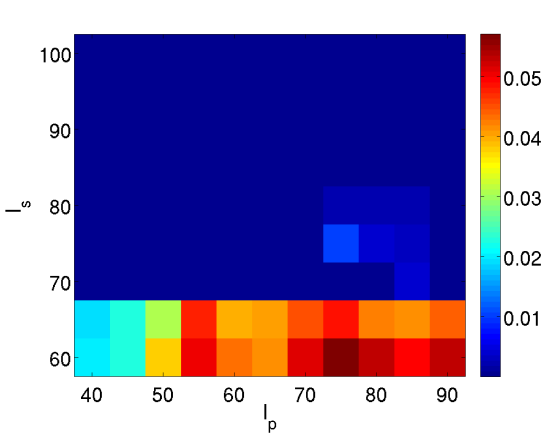
Figure 35: Amplification factor γ^i for each time step in Case 2

of $\max_i \gamma^i$ for different combinations of l_p and l_S for Galerkin (lower left) and Petrov-Galerkin (lower right) projection methods for Case 2. These plots will be referred to as stability maps. Note that the limits of the color bars for the two plots are very different (the minimum value for each color bar, which corresponds to the minimum value in the corresponding map, is indicated). As discussed earlier, for a problem with a constant matrix $\mathbf{M}$ in which time tends to infinity, we would require $\log_{10}(\max_i \gamma^i) < 0$ to assure stability. However, in practice, stable behavior in POD-TPWL models is observed as long as $\log_{10}(\max_i \gamma^i)$ is close to zero (e.g., $\log_{10}(\max_i \gamma^i) \lesssim 0.02$ for our examples; this precise value may be problem/control-setting dependent). This is the case for two reasons. First, because our models entail $O(100)$ time steps, very small growth rates do not lead to unbounded errors. Second, because the amplification matrix ($\mathbf{M}^{i+1}$ or $\mathbf{M}_r^{i+1}$) varies from time step to time step, and because Figure 36 depicts the maximum $\log_{10} \gamma^i$ over all time steps, the requirement that $\log_{10}(\max_i \gamma^i)$ be less than zero is overly strict. In the following, we thus refer to cases for which $\max_i \gamma^i < 1.05$, which corresponds to $\log_{10}(\max_i \gamma^i) < 0.021$, as behaving stably.

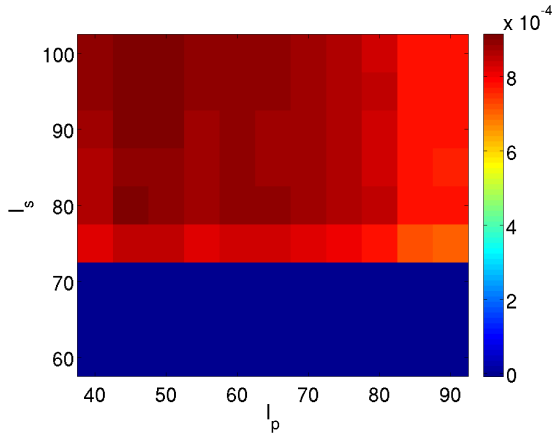
The $\log_{10}(\max_i \gamma^i)$ for Galerkin projection for Case 2 is displayed in Figure 36c. It can be seen from the plots that for Galerkin projection POD-TPWL stability is very sensitive to the choice of (l_p, l_S), and there is no clear trend. Many of the l_p and l_S combinations lead to instability (as they correspond to large $\max_i \gamma^i$), while there are some l_p and l_S combinations for which the POD-TPWL model should behave stably. On the other hand, as shown in Figure 36d, for Petrov-Galerkin projection the value of $\log_{10}(\max_i \gamma^i)$ for any combination of l_p and l_S is very close to zero (the largest value being 0.0004). Thus this method behaves stably for all combinations of l_p and l_S considered.

Also shown in Figure 36 are the stability maps for Galerkin (upper left) and Petrov-Galerkin (upper right) for Case 1 (equal phase densities). It is clear that for Case 1, Petrov-Galerkin

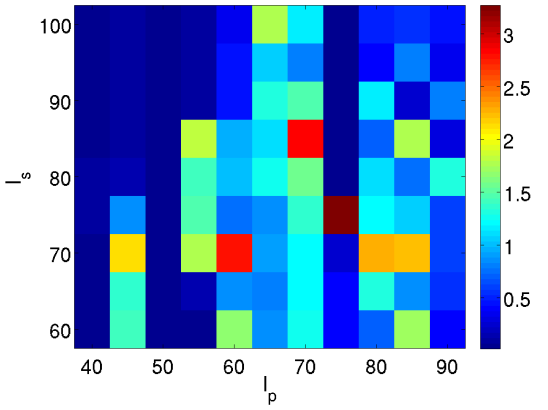
projection should behave stably for all (l_p, l_S) considered, and Galerkin projection should behave stably over a large range of l_p and l_S. Comparison of Figures 36a and 36c demonstrates that the inclusion of more complicated physics has the potential to adversely affect the performance of POD-TPWL models that apply Galerkin projection for constraint reduction. This problem may be related to the fact that the Jacobian matrix is not SPD, as discussed in Section A.5.2.



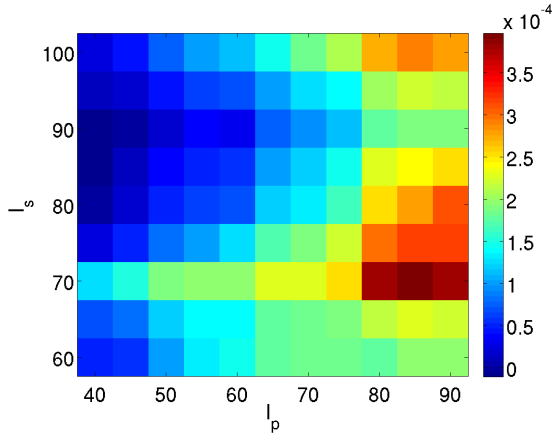
(a) Galerkin projection for Case 1 (minimum value on color bar: 4.8×10^{-6})



(b) Petrov-Galerkin projection for Case 1 (minimum value on color bar: -8.0×10^{-6})



(c) Galerkin projection for Case 2 (minimum value on color bar: 1.7×10^{-2})



(d) Petrov-Galerkin projection for Case 2 (minimum value on color bar: -1.1×10^{-5})

Figure 36: Maps of $\log_{10}(\max_i \gamma^i)$ for Cases 1 and 2

Because the stability characteristics of POD-TPWL models that apply Galerkin projection can be sensitive to the choice of l_p and l_S, the model can be 'stabilized' (in a practical sense) through the careful selection of l_p and l_S, assuming there exists an (l_p, l_S) combination with $\log_{10}(\max_i \gamma^i) < 0.021$. Such an approach was applied in [41]. For the current example, for $l_p = 75$ and $l_S = 80$, the value of $\log_{10}(\max_i \gamma^i)$ is below this threshold (see Figure 36c), which suggests that the resulting POD-TPWL model will behave stably. Figure 37 shows the water injection rate for Injector 1 for both projection methods with $l_p = 75$ and $l_S = 80$. It is clear that the results using Galerkin projection no longer exhibit fluctuations (compare Figures 34 and 37), and that the

Petro-Galerkin results continue to be stable, as would be expected from Figure 36d. These results illustrate the efficacy of constructing the POD-TPWL stability maps shown in Figure 36.

Table 3 shows the error for the Petrov-Galerkin method and the stably-behaving Galerkin method. The results indicate that the stabilized Galerkin projection is more accurate for this case (with the exception of E_S), which is consistent with the observations for Case 1.

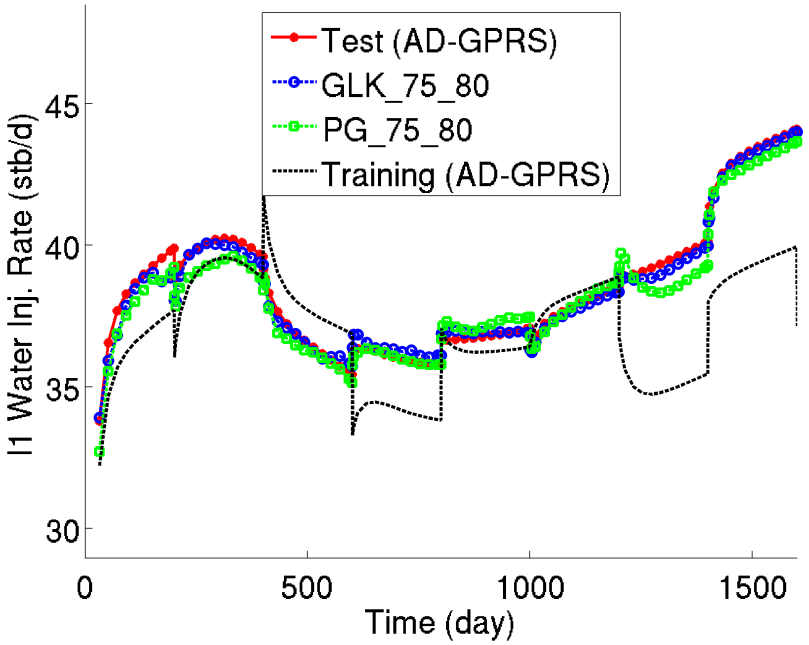


Figure 37: Water injection rate for Injector 1 in Case 2, with l_p and l_S selected based on Figure 36c

Table 3: Summary of error for Case 2

	E_o	E_w	E_{iw}	E_p (psi)	E_S
GLK_75_80	0.0216	0.0152	0.0050	13.6	0.00212
PG_75_80	0.0225	0.0255	0.0112	15.0	0.00204

A.7.5 Case 3: Compositional Simulation

Case 3 involves a four-component system in which we model the injection of CO_2 into an oil reservoir. The original fluid in place, in terms of overall mole fractions, consists of 0.01 CO_2, 0.11 of the C_1 component, 0.29 of the C_4 component, and 0.59 of the C_{10} component. Pure CO_2 is injected. The reservoir model for this case is defined on a $32 \times 40 \times 8$ grid, which translates to 10,240 grid blocks and thus 40,960 primary variables ($10{,}240 \times 4$). The permeability field, shown in Figure 38, was generated geostatistically using sequential Gaussian simulation within the SGeMS geological modeling package [61]. The model contains four producers at the four corners and one injector in the middle, forming a five-spot pattern. The producers are perforated in the lower four layers and the injector in the upper four layers.

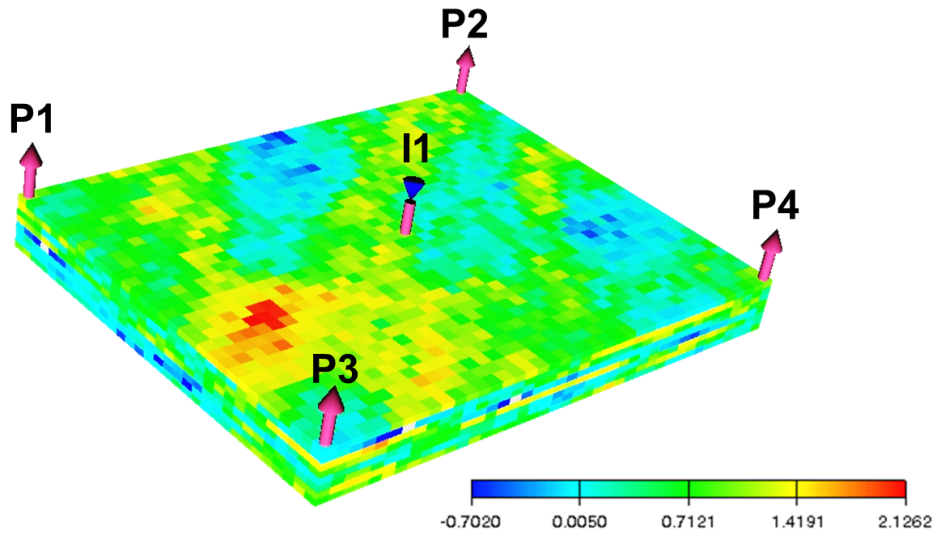
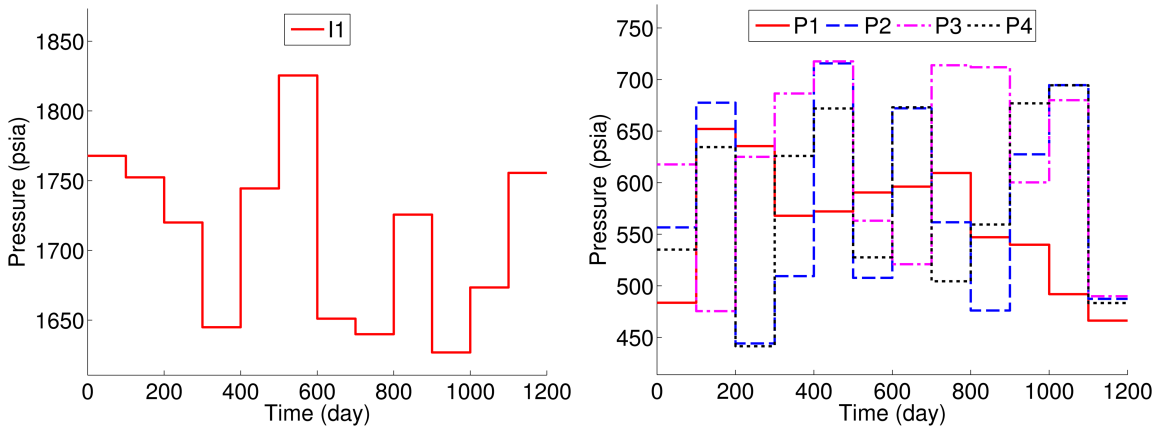


Figure 38: Reservoir model for Case 3 (log-permeability is shown)

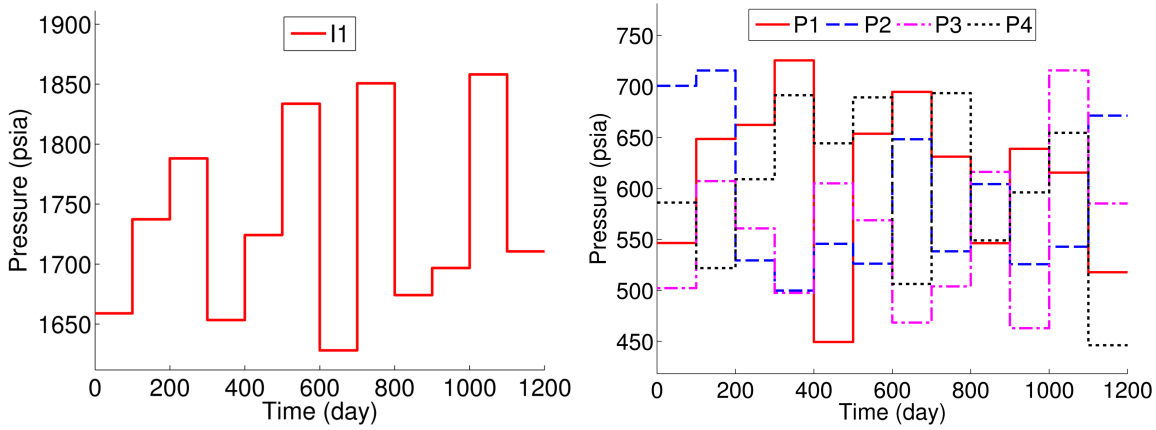
We perform two training runs, one of which is the primary training simulation, to construct the POD-TPWL model. The BHP controls for the injector and four producers are shown in Figure 39. These BHPs are varied every 100 days and are generated randomly. From the two training simulations, 275 snapshots are collected to build the basis matrix $\mathbf{\Phi}$. We specify $l_p = 120$ and $l_z = 150$. Figure 40 shows the BHPs for the test case, which are also randomly generated.



(a) Injector BHPs (b) Producer BHPs

Figure 39: Time-varying BHPs for the primary training simulation for Case 3

Figure 41 shows the stability maps for the Galerkin and Petrov-Galerkin methods for this case. Note that the scales of the color bars for the two figures are very different. For Petrov-Galerkin projection, all (l_p, l_z) combinations lead to models that behave stably. For Galerkin projection, for all (l_p, l_z) combinations considered, $\log_{10}(\max_i \gamma^i)$ is always much larger than 0 (the lowest $\max_i \gamma^i$ value being about 27). This means that for any of these (l_p, l_z) combinations, the resulting POD-TPWL model will have one or more unstable steps. Therefore, for this case, we cannot apply

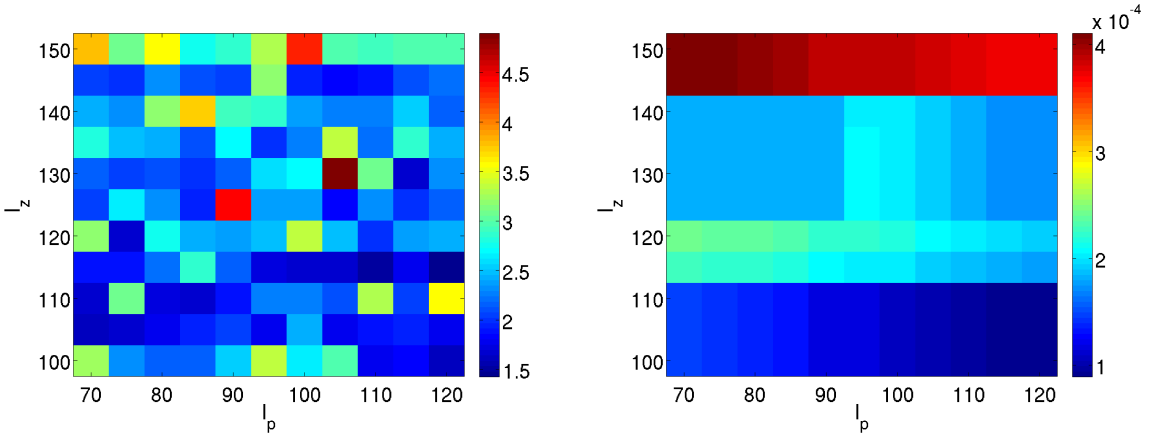


(a) Injector BHPs (b) Producer BHPs

Figure 40: Time-varying BHPs for the test simulation for Case 3

the basis selection procedure used for Case 2 to construct a stably-behaving POD-TPWL model using Galerkin projection.

In addition, although not shown here, for all (l_p, l_z) combinations considered, the Galerkin method displays $\log_{10}(\max_i \gamma^i) > 0$ for all time steps. The reason why compositional systems are less stable than oil-water systems with Galerkin projection is not entirely clear, but it may be due, at least in part, to the large density difference between the oil and gas phases (which leads to strong gravity-driven countercurrent flow) and to the high nonlinearity resulting from complex phase behavior. This issue should be investigated in future work.



(a) Galerkin projection (minimum value on color bar: 1.43)

(b) Petrov-Galerkin projection (minimum value on color bar: 8.9×10^{-5})

Figure 41: Maps of $\log_{10}(\max_i \gamma^i)$ for Case 3

Figures 42 and 43 show the oil and gas production rates for Producer 1, while Figure 44 displays the gas injection rate for Injector 1. Test-case results using Petrov-Galerkin projection are shown along with results using weighted inverse projection, which will be discussed in the next section.

Only results for the first 600 days of the simulation are shown. This allows us to focus on the period where gas production rates are increasing and the most error occurs. The POD-TPWL model exhibits reasonable overall accuracy, though errors are evident for different quantities at different times. For example, the oil production rate in Figure 42 displays some inaccuracy from 300–400 days, as do gas injection rates (Figure 44) over the first 200 days. The solution, however, clearly behaves stably, and the results are of sufficient accuracy to be useful for many applications. These results are representative of those for other wells. Runtime speedup for this case is around a factor of 150, which is lower than that for Cases 1 and 2. The lower speedup here results from the flash calculation, which is not handled efficiently in our current POD-TPWL model. This cost can, however, be significantly reduced through an efficient implementation.

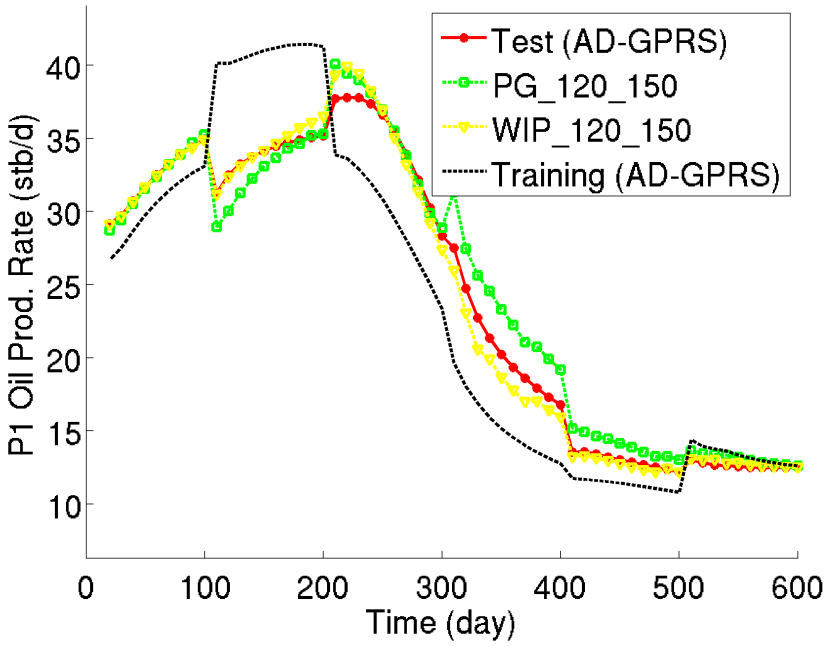


Figure 42: Oil production rate for Producer 1 in Case 3

A.8 Inverse Projection and Weighted Inverse Projection Constraint Reduction Methods

In Section A.5 we showed that, to minimize constraint reduction error $\mathbf{e}_{23}$, optimal constraint reduction methods should be of the form $\mathbf{\Psi}^T = \mathbf{\Phi}^T\mathbf{\Theta}\mathbf{J}^{-1}$. We use $\mathbf{\Theta} = \mathbf{J}$ in the case of Galerkin projection and $\mathbf{\Theta} = \mathbf{J}^T\mathbf{J}$ in the case of Petrov-Galerkin projection. Both choices eliminate the $\mathbf{J}^{-1}$ term and are thus very efficient to compute. Other choices for $\mathbf{\Theta}$ also exist, however. Though they typically entail higher computational costs, they may have theoretical advantages and display superior performance. In this section we consider two such projection methods, inverse projection and weighted inverse projection.

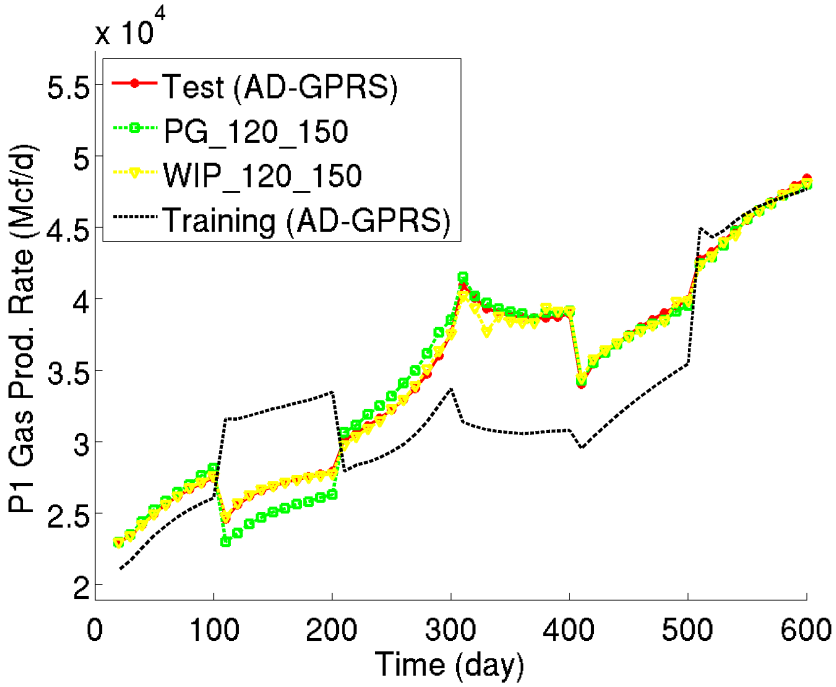


Figure 43: Gas production rate for Producer 1 in Case 3

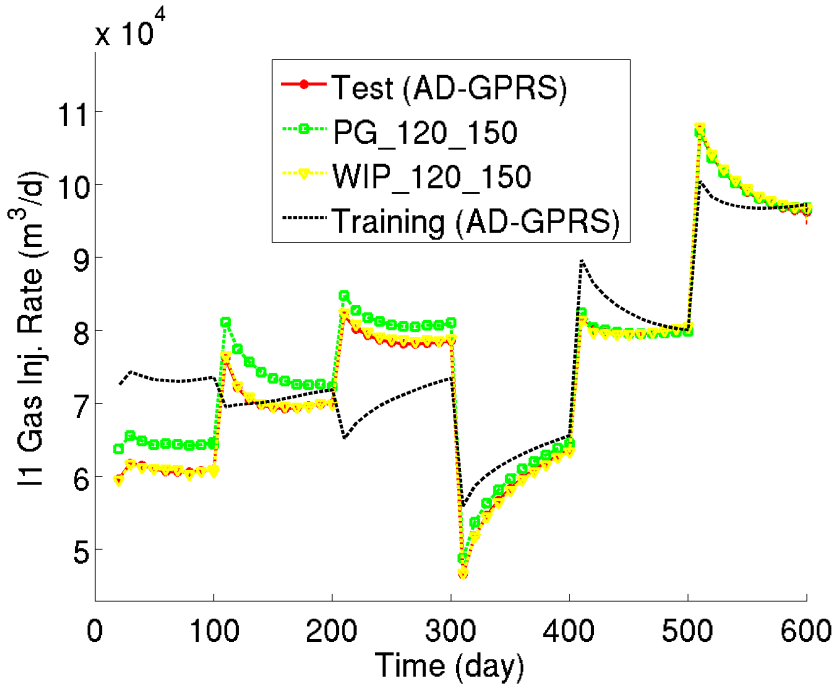


Figure 44: Gas injection rate for Injector 1 in Case 3

A.8.1 Method Development

An intuitive choice for $\mathbf{\Theta}$ is the identity matrix. Then the optimal projection matrix is given by $\mathbf{\Psi}^T = \mathbf{\Phi}^T\mathbf{J}^{-1}$. We refer to this approach as the inverse projection (IP) method.

An important property of the IP method is that it minimizes the 2-norm of the constraint reduction error $\mathbf{e}_{23}$. That is, it satisfies

$$\mathbf{\Psi}^* = \arg\min_{\mathbf{\Psi}} \|\mathbf{e}_{23}\|_2^2 . \tag{74}$$

Compared to the norms appearing in the optimality conditions for the Galerkin and Petrov-Galerkin methods, the 2-norm appearing here is preferable since it does not depend on the specific structure of $\mathbf{J}$.

In many reservoir simulation applications, such as production optimization, we are particularly interested in the states in well blocks because these values directly affect injection and production rates. We can assign larger weights to the elements of the error vector corresponding to these blocks in Equation 56 by defining $\mathbf{\Theta} = \mathbf{W}^T\mathbf{W}$, where $\mathbf{W}$ is a diagonal weighting matrix. The resulting projection matrix is $\mathbf{\Psi}^T = \mathbf{\Phi}^T\mathbf{W}^T\mathbf{W}\mathbf{J}^{-1}$. We refer to this method as weighted inverse projection (WIP). WIP degenerates to IP when $\mathbf{W} = \mathbf{I}$. WIP minimizes the weighted error vector in the 2-norm,

$$\mathbf{\Psi}^* = \arg\min_{\mathbf{\Psi}} \|\mathbf{W}\mathbf{e}_{23}\|_2^2 . \tag{75}$$

In the examples in this work involving WIP, we assign a value of 5 to the diagonal elements of $\mathbf{W}$ corresponding to well block states and 1 to the remaining diagonal elements. This 5 to 1 ratio was determined through limited numerical experimentation. The values of $\mathbf{W}$ can be further tuned to improve accuracy in particular grid blocks.

The IP and WIP methods offer theoretical advantages by virtue of their optimality conditions. Achieving this benefit, however, requires additional computational effort. This is because, for both the IP and WIP methods, the $\mathbf{J}^{-1}$ term in the expression for $\mathbf{\Psi}$ is not multiplied by $\mathbf{J}$, so it does not cancel as in the Galerkin and Petrov-Galerkin procedures.

In practice, $\mathbf{J}^{-1}$ does not need to be calculated explicitly. The matrix $\mathbf{\Psi}$ only appears in the calculation of the reduced derivatives in Equation 48. Substituting Equation 62 into Equation 48 (and dropping the superscript $i+1$) we have

$$\mathbf{J}_r = \mathbf{\Phi}^T\mathbf{\Theta}\mathbf{\Phi}, \quad \mathbf{A}_r = \mathbf{\Phi}^T\mathbf{\Theta}\mathbf{J}^{-1}\mathbf{A}\mathbf{\Phi}, \quad \mathbf{B}_r = \mathbf{\Phi}^T\mathbf{\Theta}\mathbf{J}^{-1}\mathbf{B}. \tag{76}$$

Interestingly, the calculation of $\mathbf{J}_r$ now does not involve $\mathbf{J}$. The term $\mathbf{J}^{-1}\mathbf{A}\mathbf{\Phi}$ in matrix $\mathbf{A}_r$ can be calculated by solving $\mathbf{J}\mathbf{x}_A = \mathbf{A}\mathbf{\Phi}$, which is an $n_v \times n_v$ system with l right hand sides, where l is the total number of reduced variables. Similarly, the term $\mathbf{J}^{-1}\mathbf{B}$ in matrix $\mathbf{B}_r$ can be calculated by solving $\mathbf{J}\mathbf{x}_B = \mathbf{B}$, which is an $n_v \times n_v$ system with n_u right hand sides, where n_u is the dimension of the control parameter $\mathbf{u}$. In total, we will thus need to solve the high-dimensional linear system with $l + n_u$ right hand sides at each time step. For a case with $l = 200$ and $n_u = 10$, the cost

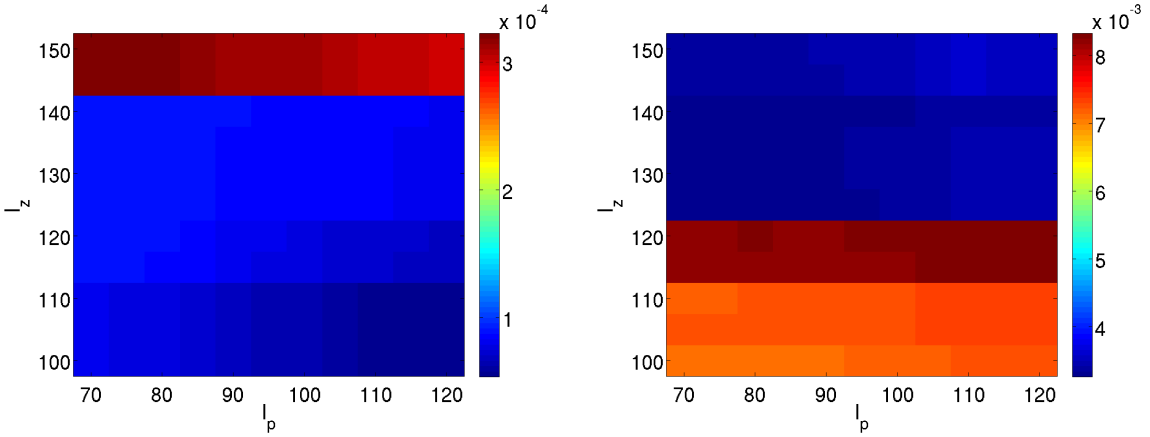
of constructing $\mathbf{A}_r$ and $\mathbf{B}_r$ for the IP or WIP method is the equivalent of about 10–20 full-order simulations for a compositional problem and 20–40 full-order simulations for an oil-water problem (this difference arises because full-order compositional simulations usually require more Newton iterations and time-step cuts than oil-water problems). Preprocessing for IP and WIP is thus much more expensive than for Galerkin and Petrov-Galerkin methods (though runtimes, and thus runtime speedup, are comparable). However, in applications where the POD-TPWL model can be used in place of hundreds of full-order runs, the IP and WIP methods may be viable options.

We note finally that the computational cost of IP and WIP may be reduced significantly by approximating $\mathbf{J}^{-1}$ with another matrix $\mathbf{Q}$, so that we can use $\mathbf{Q}$ in place of $\mathbf{J}^{-1}$. Potential choices for $\mathbf{Q}$ are the various preconditioners used for solving $\mathbf{Jx} = \mathbf{b}$. This should be considered in future work.

A.8.2 Numerical Results using IP and WIP

The IP and WIP methods will now be applied to Cases 1–3. The problem setups are identical to those described earlier.

In terms of stability, the IP and WIP methods behave stably for all three cases, and their stability characteristics are not sensitive to the choice of l_p and l_S (or l_z). As an example, Figure 45 shows the stability maps for the IP and WIP methods for Case 3, which was the most challenging case for the Galerkin method. It can be seen that for both methods, $\log_{10}(\max_i \gamma^i)$ is very close to zero for all combinations of l_p and l_z considered.



(a) IP method (minimum value on color bar: 5.4×10^{-5})

(b) WIP method (minimum value on color bar: 3.3×10^{-3})

Figure 45: Maps of $\log_{10}(\max_i \gamma^i)$ for IP and WIP for Case 3

To assess their accuracy, results using IP and WIP are now compared to those using the Galerkin and Petrov-Galerkin methods for Cases 1–3. Table 4 summarizes the errors for Case 1. In this case, the injection and production rate results using IP are less accurate than those using the Galerkin method. However, the IP method provides better results in terms of the average state error for both

pressure and water saturation. This is because the IP method is, by design, optimal in minimizing the constraint reduction error globally, rather than at the well blocks. WIP, which places additional weight on the well blocks, improves upon the IP injection and production rate results (especially for E_{iw}) while maintaining state errors similar to those of the IP method.

Table 4: Summary of error for Case 1, with IP and WIP

	E_o	E_w	E_{iw}	E_p (psi)	E_S
GLK_70_100	0.0054	0.0058	0.0015	1.57	0.00066
PG_70_100	0.0113	0.0156	0.0052	7.07	0.00130
IP_70_100	0.0059	0.0064	0.0044	0.62	0.00057
WIP_70_100	0.0050	0.0062	0.0013	0.58	0.00060

Tables 5 and 6 present the errors for Cases 2 and 3. For Case 2, the IP method provides more accurate results than the Galerkin and Petrov-Galerkin methods in all five metrics. The WIP method provides the most accurate well rate predictions among all methods tested, and leads to average state errors that are again comparable to those of the IP method. For Case 3, the IP and WIP methods continue to provide more accurate results than those obtained using Petrov-Galerkin projection.

Table 5: Summary of error for Case 2, with IP and WIP

	E_o	E_w	E_{iw}	E_p (psi)	E_S
GLK_75_80	0.0216	0.0152	0.0050	13.6	0.00212
PG_75_80	0.0225	0.0255	0.0112	15.0	0.00204
IP_75_80	0.0058	0.0077	0.0041	1.43	0.00084
WIP_75_80	0.0055	0.0076	0.0013	0.759	0.00085

Table 6: Summary of error for Case 3, with IP and WIP

	E_o	E_g	E_{ig}	E_p (psi)	E_z
PG_120_150	0.0400	0.0160	0.0187	5.68	0.00894
IP_120_150	0.0239	0.0103	0.0081	1.50	0.00393
WIP_120_150	0.0219	0.0085	0.0077	1.41	0.00395

Figures 42–44 display the performance of the WIP method relative to the Petrov-Galerkin method for Case 3. It is clear that WIP leads to improved results compared to Petrov-Galerkin projection, most notably in oil production rate from 300–400 days, in gas production rate from 100–200 days, and in gas injection rate before 300 days.

The results in this section indicate that the use of (weighted) inverse projection for constraint reduction leads to POD-TPWL models that are more accurate than those generated using Galerkin or Petrov-Galerkin projection. In addition, in common with Petrov-Galerkin projection, IP and WIP provide POD-TPWL models that behave stably. The IP and WIP methods, as currently implemented, require much more preprocessing than the Petrov-Galerkin procedure, so the overall

level of speedup they provide is limited. Future work should target the fast construction of the reduced matrices $\mathbf{A}_r$ and $\mathbf{B}_r$. Progress in this direction would enable the efficient use of IP and WIP for large models.

A.9 Concluding Remarks

In this Appendix we assessed the various errors that arise in POD-TPWL models. These errors derive from state reduction, nonlinearity treatment (accomplished here using piecewise linearization) and constraint reduction. Our focus was on constraint reduction, which is applied to project the overdetermined high-dimensional system of equations into a low-dimensional subspace. Once constraint reduction and state reduction (accomplished here through use of POD) are applied, the low-dimensional set of equations is fully determined. The choice of constraint reduction procedure was shown to affect POD-TPWL error and stability behavior. In previous work on the use of POD-TPWL for subsurface flow, Galerkin projection has mainly been applied, though recent work has used Petrov-Galerkin projection for compositional reservoir simulation [39]. Here we showed that both the Galerkin and Petrov-Galerkin projection methods can be derived from the optimality condition of the constraint reduction error, though the Galerkin method does not satisfy the optimality condition when the Jacobian matrix is not SPD (which it is not in oil-water or compositional flow problems).

The performance of the Galerkin and Petrov-Galerkin methods was compared for two oil-water cases and one oil-gas compositional case. For oil-water systems with gravity-driven countercurrent flow, it was observed that the stability behavior of Galerkin projection is sensitive to the number of reduced variables (l_p, l_S) used. In these cases, it may however be possible to find an (l_p, l_S) combination for which Galerkin projection behaves stably. For compositional simulation, the Galerkin method was unstable for any of the (l_p, l_z) combinations tested. By contrast, the Petrov-Galerkin method was shown to behave stably for all cases considered (both here and in [39]). In addition, its stability characteristics were not found to be sensitive to the number of reduced variables used. Therefore, for complex reservoir simulation problems, Petrov-Galerkin projection appears to be more reliable compared to Galerkin projection. This observation is consistent with the findings in [19], where Galerkin and Petrov-Galerkin projection were compared for reduced-order models for turbulent flow and nonlinear structural dynamics problems. We note, however, that theoretical guarantees regarding the stability of Petrov-Galerkin projection for nonlinear problems do not, to our knowledge, exist.

There are other methods that satisfy the optimality conditions for constraint reduction, and two such procedures were considered. Inverse projection (IP) minimizes the 2-norm of the constraint reduction error, while weighted inverse projection (WIP) minimizes the 2-norm of the error vector with extra weight at selected blocks. The WIP method is by design optimal for well flow rate calculations (when well blocks are weighted more heavily), and it was shown to provide the best overall accuracy among all of the constraint reduction methods considered. However, because they involve the inverse Jacobian matrix $\mathbf{J}^{-1}$, the IP and WIP methods incur high computational costs

for the construction of the POD-TPWL model. Specifically, preprocessing (overhead) costs for these methods are equivalent to about 10–40 full-order simulations. Improving the efficiency of these methods should be a topic for future research.

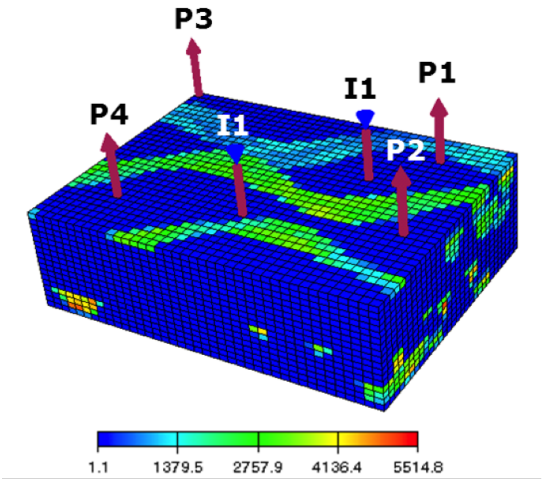
Although our specific focus in this Appendix was on subsurface flow, a variety of problems are governed by conservation laws similar to Equation 36, so many of our detailed findings should be more broadly relevant. In addition, because most aspects of the development presented in this Appendix are not specific to a particular application, we expect our general approaches and findings to be applicable to a range of problems modeled using POD-TPWL. More specifically, the step-by-step error assessment, optimality results for the various constraint reduction treatments, stability analysis, and the procedure for low-dimensional stability map construction, should all be applicable for POD-TPWL methods in general. Our findings regarding the stability advantages of Petrov-Galerkin projection relative to Galerkin projection should hold for other problems in which the Jacobian matrix is not SPD. We also believe that the IP and WIP approaches for constraint reduction may provide higher accuracy for a range of applications (though they require efficiency improvements, as noted above). A few treatments are, however, application specific. These include the selection of the number of training runs and the controls applied in these runs, the number of snapshots required, the detailed construction of the basis matrix $\mathbf{\Phi}$, and the specific definition of 'distance' used to determine the nearest saved state.

Future work should consider the development of constraint reduction methods that are optimally accurate and guaranteed to be stable by, for example, combining the optimality condition presented in this work with the Lyapunov stability equation. The quantification and reduction of the other errors that arise in POD-TPWL models should also be addressed. The linearized treatment could conceivably be improved by incorporating (or estimating) higher-order corrections at selected locations and times. For subsurface flow applications it will additionally be of interest to consider cases with more wells (e.g., 10–100). With larger numbers of wells, there is more variability in the states that can occur in the model. This suggests that more snapshots, from more training runs, will be needed to represent the system. The development of techniques for the systematic design of training runs may enable the POD-TPWL model to provide sufficient accuracy, with reasonable overhead, for practical cases.

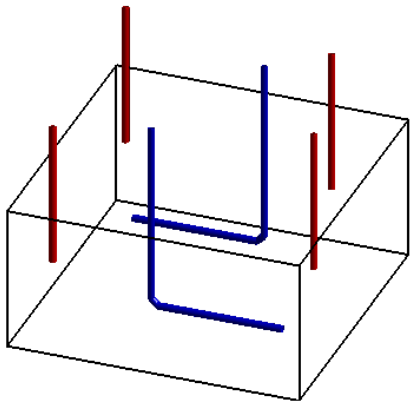
B POD-TPWL for CO_2 EOR

The emphasis of this research has been on the application of POD-TPWL for CO_2-water systems. However, we also tested the general oil-gas compositional POD-TPWL model for an example involving CO_2 EOR. Such simulations are relevant to carbon capture, utilization and storage (CCUS), in which CO_2 is used for oil recovery as it is being sequestered. In our model here, CO_2 injection is accomplished using horizontal wells. The POD-TPWL method used in this example is that of [39]. In that work, POD-TPWL modeling of CO_2 EOR was presented, but horizontal wells were not considered.

The geological model used here is based on the Stanford VI (channelized model) [20], which was also considered in Section 3. The simulation model here is taken directly from [39] — only the wells were modified. The model, which contains $30 \times 40 \times 17$ cells (total of 20,400 grid blocks) and is of overall size 585 m $\times$ 780 m $\times$ 17 m, is shown in Fig. 46a. There are six wells in the reservoir – four vertical producers and two horizontal injectors (see Fig. 46). The two injectors are of length 292 m and 370 m and they are located in the bottom layer of the model. The production wells penetrate the top five layers of the model.



(a) Geological model



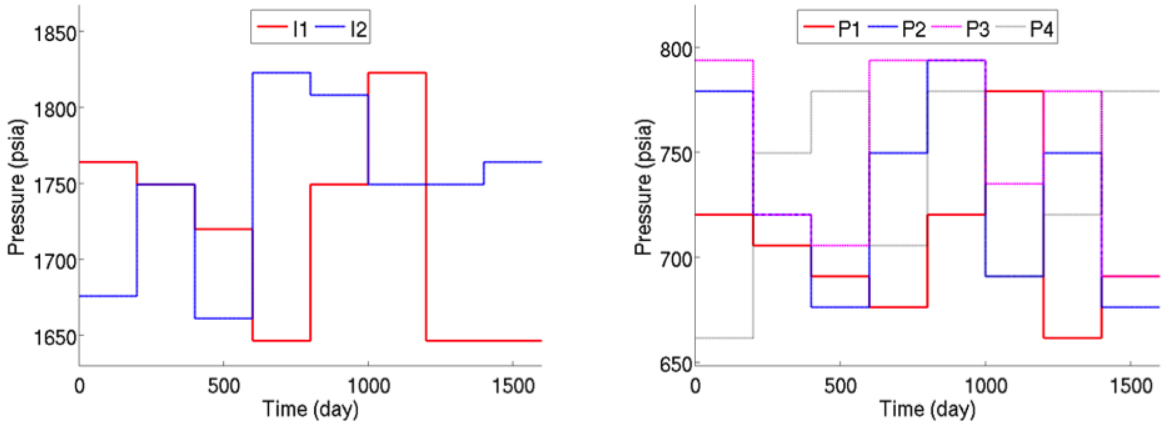
(b) Well locations

Figure 46: Geological model and well locations (from [20, 39])

The model includes four components (CO_2, C_1, C_4, C_{10}) in oil and gas phases (water is not included). Thus the simulation model contains 81,600 primary variables ($20,400 \times 4$). The initial reservoir pressure is 100 bar at the top of the model. The overall molar fractions at the start of the simulation are 0.01 CO_2, 0.11 C_1, 0.29 C_4, and 0.59 C_{10}. CO_2 of purity 97% is injected at both injectors (injected fluid also contains 1% C_1, 1% C_4, 1% C_{10}). The wells are BHP controlled and the model is run for 1600 days. All other model properties are as in [39].

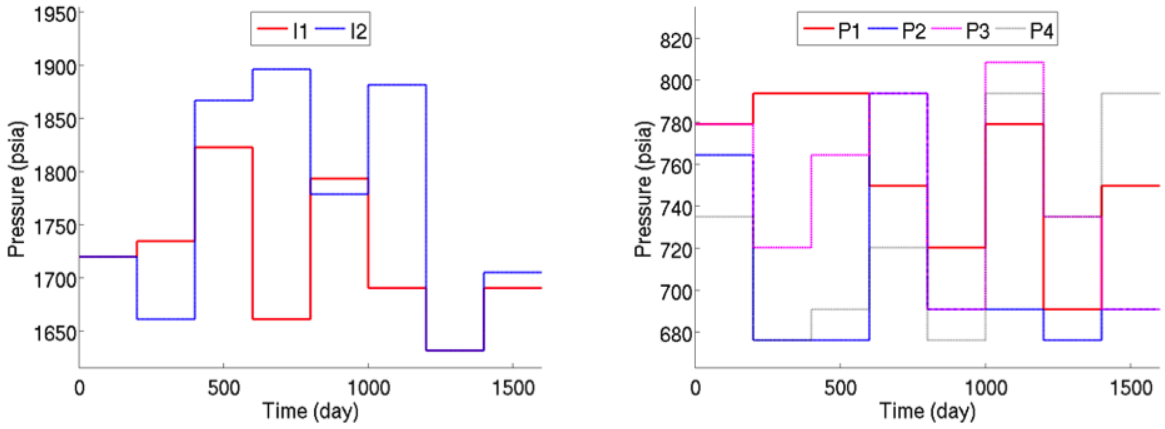
Two training cases were simulated in order to construct the POD-TPWL model. The BHP schedules for these runs are shown in Figs. 47 and 48. A total of 362 snapshots were generated. In

the POD-TPWL model, we use $l_p = 90$ and $l_z = 250$. The test-case BHPs are shown in Fig. 49. These BHPs clearly differ from those for the training runs, but their ranges are similar. As in Section 3, these BHPs are intended to qualitatively resemble the profiles evaluated during the course of an optimization run.



(a) Injection well BHPs (b) Production well BHPs

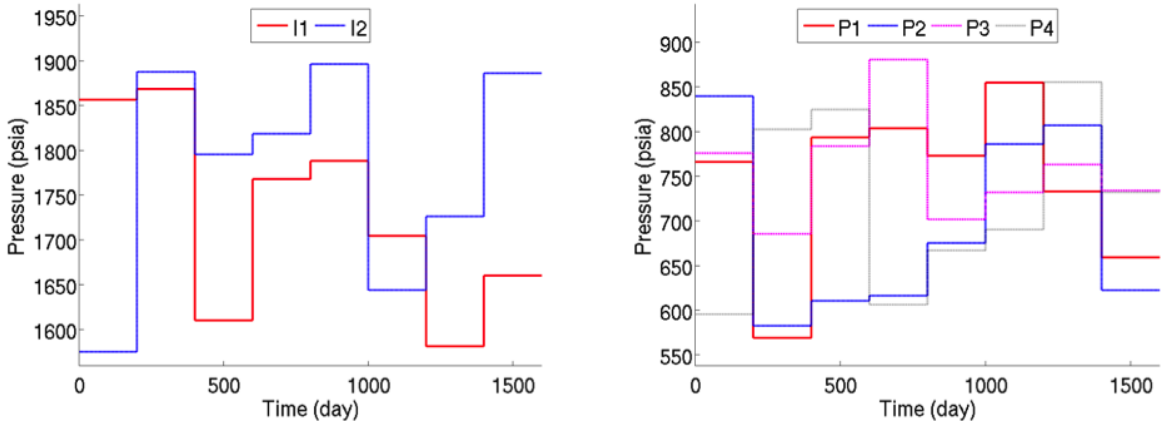
Figure 47: Time-varying BHPs for training case 1



(a) Injection well BHPs (b) Production well BHPs

Figure 48: Time-varying BHPs for training case 2

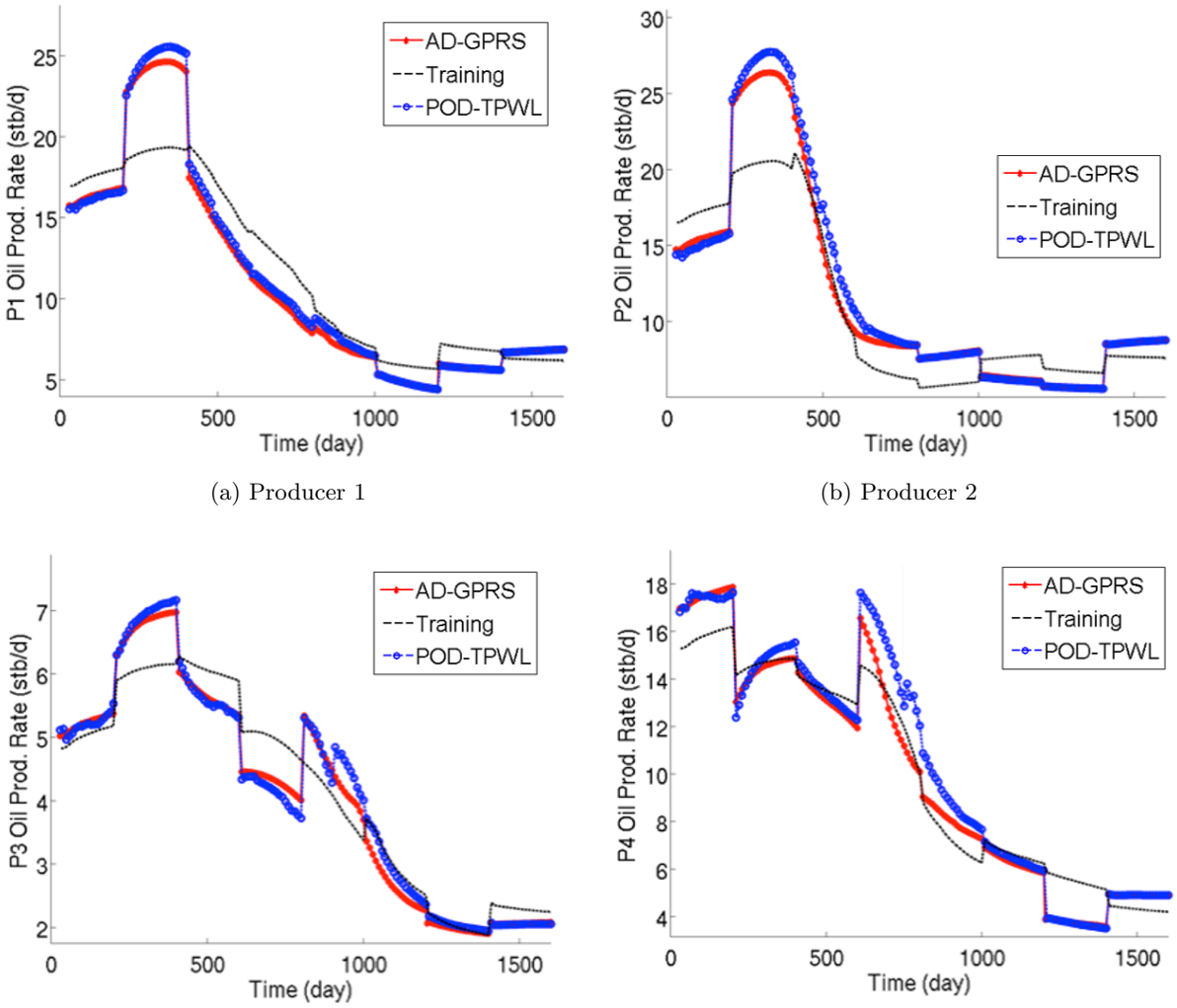
POD-TPWL results for oil and gas production rates are shown in Figs. 50 and 51. Injection rates for the two horizontal CO_2 injectors are shown in Fig. 52. The various curves are as described in Section 3, and it is apparent that there are substantial differences between training and test-case results for this example. We observe that the POD-TPWL model provides results in reasonably close agreement with the reference AD-GPRS results. Some discrepancies are noticeable, but the overall trends are captured quite well. In addition, the general level of accuracy appears to be



(a) Injection well BHPs (b) Production well BHPs

Figure 49: Time-varying BHPs for test case

comparable to that in [39], where only vertical wells were modeled. This is a useful observation, as these are the first POD-TPWL CO_2 EOR runs that include horizontal wells.



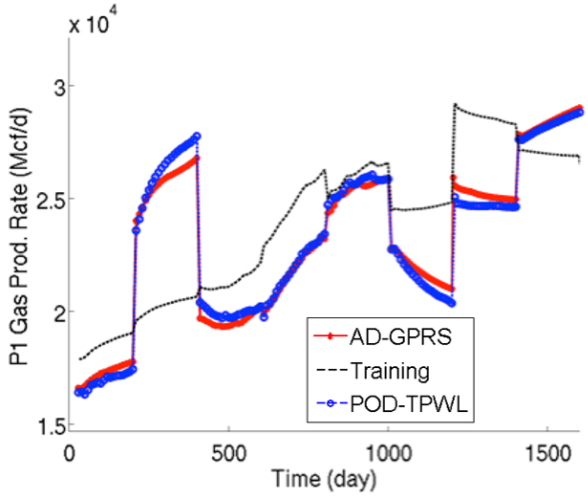
(a) Producer 1

(b) Producer 2

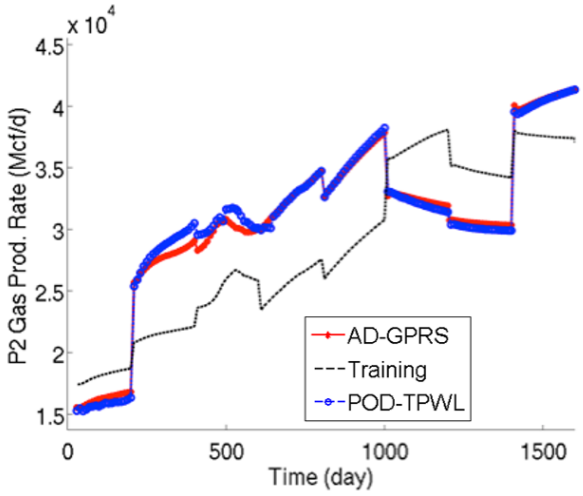
(c) Producer 3

(d) Producer 4

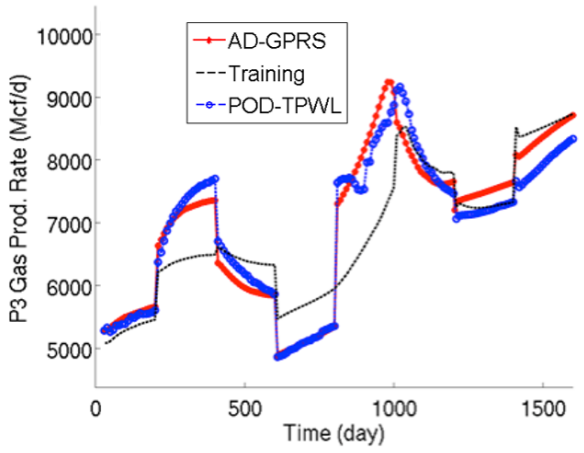
Figure 50: Oil production rates



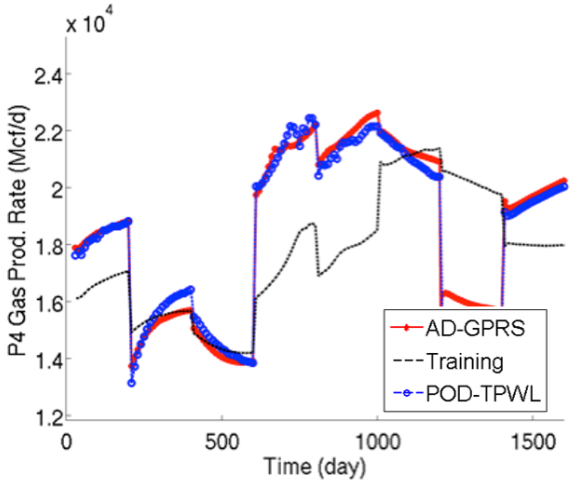
(a) Producer 1



(b) Producer 2

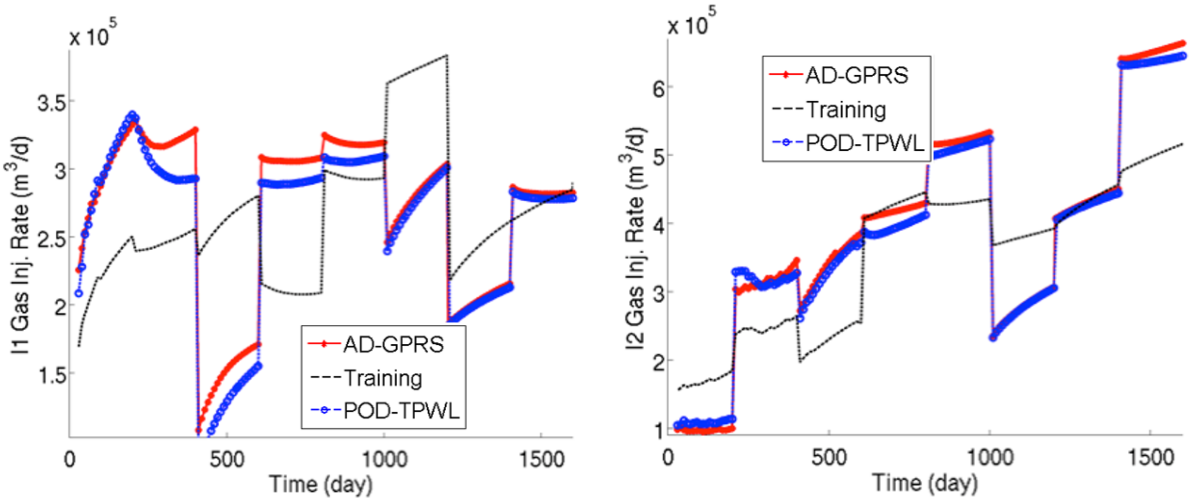


(c) Producer 3



(d) Producer 4

Figure 51: Gas production rates



(a) Injector 1 (b) Injector 2

Figure 52: CO_2 injection rates

References

[1] S. Anbar. Development of a predictive model for carbon dioxide sequestration in deep saline carbonate aquifers (SPE paper 141135). In *SPE Annual Technical Conference and Exhibition*, Florence, Italy, September 2010.

[2] R. J. Arts, A. Chadwick, O. Eiken, S. Thibeau, and S. Nooner. Ten years' experience of monitoring CO_2 injection in the Utsira Sand at Sleipner, offshore Norway. *First Break*, 26(1), 2008.

[3] K. Aziz and A. Settari. *Fundamentals of Reservoir Simulation.* Elsevier Applied Science Publishers, 1986.

[4] G. Berkooz and E. Z. Titi. Galerkin projections and the proper orthogonal decomposition for equivariant equations. *Physics Letters A*, 174(1-2):94–102, 1993.

[5] B. N. Bond and L. Daniel. Guaranteed stable projection-based model reduction for indefinite and unstable linear systems. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 728–735, San Jose, California, USA, 2008.

[6] B. N. Bond and L. Daniel. Stable reduced models for nonlinear descriptor systems through piecewise-linear approximation and projection. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(10):1467–1480, 2009.

[7] A. Bonneville et al. Evaluating the suitability for CO_2 storage at the FutureGen 2.0 Site, Morgan County, Illinois, USA. *Energy Procedia*, 37:6125–6132, 2013.

[8] M. E. Boot-Handford et al. Carbon capture and storage update. *Energy & Environmental Science*, 7(1):130–189, 2014.

[9] T. Bui-Thanh, M. Damodaran, and K. Willcox. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal*, 42(8):1505–1516, 2004.

[10] T. Bui-Thanh, K. Willcox, and O. Ghattas. Model reduction for large-scale systems with high-dimensional parametric input space. *SIAM Journal on Scientific Computing*, 30(6):3270–3288, 2008.

[11] M. Burton, N. Kumar, and S. L. Bryant. Time-dependent injectivity during CO_2 storage in aquifers (SPE paper 113937). In *SPE Symposium on Improved Oil Recovery*, Tulsa, Oklahoma, USA, April 2008.

[12] L. Cai and R. E. White. Reduction of model order based on proper orthogonal decomposition for lithium-ion battery simulations. *Journal of the Electrochemical Society*, 156(3):A154–A161, 2009.

[13] D. A. Cameron and L. J. Durlofsky. Optimization of well placement, CO_2 injection rates, and brine cycling for geological carbon sequestration. *International Journal of Greenhouse Gas Control*, 10:100–112, 2012.

[14] D. A. Cameron and L. J. Durlofsky. Optimization and data assimilation for geological carbon storage. In J. Bundschuh and R. Al-Khoury, editors, *Computational Models for CO_2 Geo-sequestration & Compressed Air Energy Storage*, pages 357–388. CRC Press, 2014.

[15] Y. Cao, J. Zhu, I. M. Navon, and Z. Luo. A reduced-order approach to four-dimensional variational data assimilation using proper orthogonal decomposition. *International Journal for Numerical Methods in Fluids*, 53(10):1571–1583, 2007.

[16] M. A. Cardoso and L. J. Durlofsky. Linearized reduced-order models for subsurface flow simulation. *Journal of Computational Physics*, 229(3):681–700, 2010.

[17] M. A. Cardoso and L. J. Durlofsky. Use of reduced-order modeling procedures for production optimization. *SPE Journal*, 15(2):426–435, 2010.

[18] M. A. Cardoso, L. J. Durlofsky, and P. Sarma. Development and application of reduced-order modeling procedures for subsurface flow simulation. *International Journal for Numerical Methods in Engineering*, 77(9):1322–1350, 2009.

[19] K. Carlberg, C. Bou-Mosleh, and C. Farhat. Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2):155–181, 2011.

[20] S. A. Castro. *A Probabilistic Approach to Jointly Integrate 3D/4D Seismic, Production Data and Geological Information for Building Reservoir Models.* PhD thesis, Stanford University, 2007.

[21] S. Chaturantabut and D. C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[22] S. Chaturantabut and D. C. Sorensen. Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media. *Mathematical and Computer Modelling of Dynamical Systems*, 17(4):337–353, 2011.

[23] S. Chaturantabut and D. C. Sorensen. A state space error estimate for POD-DEIM nonlinear model reduction. *SIAM Journal on Numerical Analysis*, 50(1):46–63, 2012.

[24] Y. Chen and J. White. A quadratic method for nonlinear model order reduction. In *Proceedings of the International Conference on Modeling and Simulation of Microsystems*, pages 477–480, 2000.

[25] E. W. Cheney and D. R. Kincaid. *Numerical Mathematics and Computing.* Cengage Learning, 2012.

[26] H. Class et al. A benchmark study on problems related to CO_2 storage in geologic formations. *Computational Geosciences*, 13(4):409–434, 2009.

[27] K. H. Coats. An equation of state compositional model. *SPE Journal*, 20(5):363–376, 1980.

[28] M. Condon and R. Ivanov. Empirical balanced truncation of nonlinear systems. *Journal of Nonlinear Science*, 14(5):405–414, 2004.

[29] Y. Fan, L. J. Durlofsky, and H. A. Tchelepi. A fully-coupled flow-reactive-transport formulation based on element conservation, with application to CO_2 storage simulations. *Advances in Water Resources*, 42:47–61, 2012.

[30] S. F. Farshidi, Y. Fan, L. J. Durlofsky, and H. A. Tchelepi. Chemical reaction modeling in a compositional reservoir-simulation framework (SPE paper 163677). In *SPE Reservoir Simulation Symposium*, The Woodlands, Texas, USA, February 2013.

[31] M. Fragoso, B. Horowitz, and J. R. P. Rodrigues. Retraining criteria for TPWL/POD surrogate based waterflooding optimization (SPE paper 173252). In *SPE Reservoir Simulation Symposium*, Houston, Texas, USA, February 2015.

[32] K. Gallivan, E. Grimme, and P. V. Dooren. A rational Lanczos algorithm for model reduction. *Numerical Algorithms*, 12(1):33–63, 1996.

[33] S. E. Gasda, J. M. Nordbotten, and M. A. Celia. Vertical equilibrium with sub-scale analytical methods for geological CO_2 sequestration. *Computational Geosciences*, 13(4):469–481, 2009.

[34] I. Gelfand. Zur theorie der charactere der abelschen topologischen gruppen. *Mat.Sb.*, 9(51):49–50, 1941.

[35] M. G. Gerritsen and L. J. Durlofsky. Modeling fluid flow in oil reservoirs. *Annual Review of Fluid Mechanics*, 37:211–238, 2005.

[36] M. Ghasemi, Y. Yang, Y. Efendiev, E. Gildin, and V. M. Calo. Fast multiscale reservoir simulations using POD-DEIM model reduction (SPE paper 173271). In *SPE Reservoir Simulation Symposium*, Houston, Texas, USA, February 2015.

[37] E. Gildin, M. Ghasemi, A. Romanovskay, and Y. Efendiev. Nonlinear complexity reduction for fast simulation of flow in heterogeneous porous media (SPE paper 163618). In *SPE Reservoir Simulation Symposium*, The Woodlands, Texas, USA, Feb. 2013.

[38] J. He. *Reduced-Order Modeling for Oil-Water and Compositional Systems, with Application to Data Assimilation and Production Optimization.* PhD thesis, Stanford University, 2013.

[39] J. He and L. J. Durlofsky. Reduced-order modeling for compositional simulation by use of trajectory piecewise linearization. *SPE Journal*, 19(5):858–872, 2014.

[40] J. He and L. J. Durlofsky. Constraint reduction procedures for reduced-order subsurface flow models based on POD–TPWL. *International Journal for Numerical Methods in Engineering*, 2015, doi:10.1002/nme.4874.

[41] J. He, J. Sætrom, and L. J. Durlofsky. Enhanced linearized reduced-order models for subsurface flow simulation. *Journal of Computational Physics*, 230(23):8313–8341, 2011.

[42] J. He, P. Sarma, and L. J. Durlofsky. Reduced-order flow modeling and geological parameterization for ensemble-based data assimilation. *Computers & Geosciences*, 55:54–69, 2013.

[43] T. Heijn, R. Markovinović, and J. D. Jansen. Generation of low-order reservoir models using system-theoretical concepts. *SPE Journal*, 9(2):202–218, 2004.

[44] A. Kellems, S. Chaturantabut, D. C. Sorensen, and S. J. Cox. Morphologically accurate reduced order modeling of spiking neurons. *Journal of Computational Neuroscience*, 28:477–494, 2010.

[45] S. Krevor, R. Pini, B. Li, and S. M. Benson. Capillary heterogeneity trapping of CO_2 in a sandstone rock at reservoir conditions. *Geophysical Research Letters*, 38(15), 2011.

[46] S. Lall, J. E. Marsden, and S. Glavaški. A subspace approach to balanced truncation for model reduction of nonlinear control systems. *International Journal of Robust and Nonlinear Control*, 12(6):519–535, 2002.

[47] E. Liberge and A. Hamdouni. Reduced order modelling method via proper orthogonal decomposition (POD) for flow around an oscillating cylinder. *Journal of Fluids and Structures*, 26(2):292–311, 2010.

[48] R. Markovinović and J. D. Jansen. Accelerating iterative solution methods using reduced-order models as solution predictors. *International Journal for Numerical Methods in Engineering*, 68(5):525–541, 2006.

[49] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Transactions on Automatic Control*, 26:17–31, 1981.

[50] L. Nghiem, V. Shrivastava, B. Kohse, M. Hassam, and C. Yang. Simulation and optimization of trapping processes for CO_2 storage in saline aquifers. *Journal of Canadian Petroleum Technology*, 49(8):15, 2010.

[51] L. Nghiem, C. Yang, V. Shrivastava, B. Kohse, M. Hassam, and C. Card. Risk mitigation through the optimization of residual gas and solubility trapping for CO_2 storage in saline aquifers. *Energy Procedia*, 1(1):3015–3022, 2009.

[52] M. H. Noh, L. W. Lake, S. L. Bryant, and A. N. Araque-Martinez. Implications of coupling fractional flow and geochemistry for CO_2 injection in aquifers. *SPE Reservoir Evaluation & Engineering*, 10(4):406–414, 2007.

[53] J. M. Nordbotten and M. A. Celia. *Geological Storage of CO_2: Modeling Approaches for Large-Scale Simulation*. John Wiley & Sons, 2011.

[54] J. M. Nordbotten, M. A. Celia, and S. Bachu. Injection and storage of CO_2 in deep saline aquifers: Analytical solution for CO_2 plume evolution during injection. *Transport in Porous Media*, 58(3):339–360, 2005.

[55] J. M. Nordbotten, M. A. Celia, S. Bachu, and H. K. Dahle. Semianalytical solution for CO_2 leakage through an abandoned well. *Environmental Science & Technology*, 39(2):602–611, 2005.

[56] Y. D. Oruganti and S. Mishra. An improved simplified analytical model for CO_2 plume movement and pressure buildup in deep saline formations. *International Journal of Greenhouse Gas Control*, 14:49–59, 2013.

[57] K. Pruess, J. García, T. Kovscek, C. Oldenburg, J. Rutqvist, C. Steefel, and T. Xu. Code intercomparison builds confidence in numerical simulation models for geologic disposal of CO_2. *Energy*, 29(9):1431–1444, 2004.

[58] K. Pruess, T. Xu, J. Apps, and J. García. Numerical modeling of aquifer disposal of CO_2. *SPE Journal*, 8(1):49–60, 2003.

[59] M. Rathinam and L. R. Petzold. A new look at proper orthogonal decomposition. *SIAM Journal on Numerical Analysis*, 41(5):1893–1925, 2003.

[60] S. S. Ravindran. A reduced-order approach for optimal control of fluids using proper orthogonal decomposition. *International Journal for Numerical Methods in Fluids*, 34(5):425–448, 2000.

[61] N. Remy, A. Boucher, and J. Wu. *Applied Geostatistics with SGeMS: A User's Guide*. Cambridge University Press, 2008.

[62] M. Rewienski and J. White. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 22(2):155–170, 2003.

[63] M. A. H. Rousset, C. K. Huang, H. Klie, and L. J. Durlofsky. Reduced-order modeling for thermal recovery processes. *Computational Geosciences*, 18(3-4):401–415, 2014.

[64] J. Schuetter, P. R. Ganesh, and D. Mooney. Building statistical proxy models for CO_2 geologic sequestration. *Energy Procedia*, 63:3702–3714, 2014.

[65] J. Schuetter, S. Mishra, and D. Mooney. Evaluation of metamodeling techniques on a CO_2 injection simulation study. In *7th International Congress on Environmental Modelling and Software*, pages 16–19, San Diego, California, USA, June 2014.

[66] L. Sirovich. Analysis of turbulent flows by means of the empirical eigenfunctions. *Fluid Dynamics Research*, 8(1-4):85, 1991.

[67] J. F. M. van Doren, R. Markovinović, and J. D. Jansen. Reduced-order optimal control of water flooding using proper orthogonal decomposition. *Computational Geosciences*, 10(1):137–158, 2006.

[68] D. Vasilyev, M. Rewienski, and J. White. A TBR-based trajectory piecewise-linear algorithm for generating accurate low-order models for nonlinear analog circuits and MEMS. In *DAC '03: Proceedings of the 40th Conference on Design Automation*, pages 490–495, Anaheim, California, USA, 2003.

[69] D. Vasilyev, M. Rewienski, and J. White. Macromodel generation for BioMEMS components using a stabilized balanced truncation plus trajectory piecewise-linear approach. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(2):285–293, 2006.

[70] P. T. M. Vermeulen, A. W. Heemink, and C. B. M. Te Stroet. Reduced models for linear groundwater flow models using empirical orthogonal functions. *Advances in Water Resources*, 27(1):57–69, 2004.

[71] H. X. Vo and L. J. Durlofsky. A new differentiable parameterization based on principal component analysis for the low-dimensional representation of complex geological models. *Mathematical Geosciences*, 46(7):775–813, 2014.

[72] D. Voskov and H. A. Tchelepi. Comparison of nonlinear formulations for two-phase multi-component EoS based simulation. *Journal of Petroleum Science and Engineering*, 82:101–111, 2012.

[73] D. White. Monitoring CO_2 storage during EOR at the Weyburn-Midale Field. *The Leading Edge*, 28(7):838–842, 2009.

[74] K. Willcox and J. Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA journal*, 40(11):2323–2330, 2002.

[75] J. Wriedt, M. Deo, W. S. Han, and J. Lepinski. A methodology for quantifying risk and likelihood of failure for carbon dioxide injection into deep saline reservoirs. *International Journal of Greenhouse Gas Control*, 20:196–211, 2014.

[76] I. W. Wright. The In Salah gas CO_2 storage project. In *IPTC 2007: International Petroleum Technology Conference*, Dubai, U.A.E, December 2007.

[77] Y. J. Yang and K. Y. Shen. Nonlinear heat-transfer macromodeling for MEMS thermal devices. *Journal of Micromechanics and Microengineering*, 15(2):408–418, 2005.

[78] L. C. Young and R. E. Stephenson. A generalized compositional approach for reservoir simulation. *SPE Journal*, 23(5):727–742, 1983.

[79] Y. Zhou. *Parallel General-Purpose Reservoir Simulation with Coupled Reservoir Models and Multisegment Wells*. PhD thesis, Stanford University, 2012.