mathematics

MDPI

Article

Efficient Long-Term Simulation of the Heat Equation with Application in Geothermal Energy Storage

Martin Bähr [1,*] and Michael Breuß [2]

[1] German Aerospace Center (DLR), Institute of Low-Carbon Industrial Processes, Walther-Pauer-Straße 5, 03046 Cottbus, Germany
[2] Institute of Mathematics, Brandenburg University of Technology (BTU) Cottbus-Senftenberg, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany; breuss@b-tu.de
* Correspondence: martin.baehr@dlr.de

Abstract: Long-term evolutions of parabolic partial differential equations, such as the heat equation, are the subject of interest in many applications. There are several numerical solvers marking the state-of-the-art in diverse scientific fields that may be used with benefit for the numerical simulation of such long-term scenarios. We show how to adapt some of the currently most efficient numerical approaches for solving the fundamental problem of long-term linear heat evolution with internal and external boundary conditions as well as source terms. Such long-term simulations are required for the optimal dimensioning of geothermal energy storages and their profitability assessment, for which we provide a comprehensive analytical and numerical model. Implicit methods are usually considered the best choice for resolving long-term simulations of linear parabolic problems; however, in practice the efficiency of such schemes in terms of the combination of computational load and obtained accuracy may be a delicate issue, as it depends very much on the properties of the underlying model. For example, one of the challenges in long-term simulation may arise by the presence of time-dependent boundary conditions, as in our application. In order to provide both a computationally efficient and accurate enough simulation, we give a thorough discussion of the various numerical solvers along with many technical details and own adaptations. By our investigation, we focus on two largely competitive approaches for our application, namely the fast explicit diffusion method originating in image processing and an adaptation of the Krylov subspace model order reduction method. We validate our numerical findings via several experiments using synthetic and real-world data. We show that we can obtain fast and accurate long-term simulations of typical geothermal energy storage facilities. We conjecture that our techniques can be highly useful for tackling long-term heat evolution in many applications.

Keywords: heat equation; internal boundary conditions; efficient long-term evolution; fast explicit diffusion; Krylov subspace model order reduction; geothermal energy storage

MSC: 35K05; 65M06; 65Z05; 93-08; 93-10

Citation: Bähr, M.; Breuß, M. Efficient Long-Term Simulation of the Heat Equation with Application in Geothermal Energy Storage. *Mathematics* **2022**, *10*, 2309. https://doi.org/10.3390/math10132309

Academic Editors: Theodore E. Simos and Charampos Tsitouras

Received: 25 May 2022
Accepted: 26 June 2022
Published: 1 July 2022

1. Introduction

Alternatives to fossil fuel resources are becoming increasingly important. Besides an efficient generation of energy, it is also important to store it, ideally with minimal losses over long periods of time. The recent *geothermal energy storage* (GES) technology shown in Figure 1 represents a potentially very attractive approach to energy storage. The GES is realised in natural underground sites, for instance using large soil tanks. Such tanks are partially surrounded by insulating walls and, depending on their depth, soils with different heat conduction and capacity properties. It is a very cost effective technology that may be used in both new construction and refurbishment.

In contrast to classic underground energy storage where tanks are employed that are fully closed and do not interact with their environment, the technical realisation of GES is

characterised by a downwardly open heat tank. This is one of the aspects that makes the technology highly cost effective since it is not required to excavate the ground to a large degree. It is sufficient to excavate a relatively small area that represents the tank, staying thereby close to the surface. In the resulting pit, the heating pipes are installed, and one has to insulate the walls of the pit for instance by use of Styrofoam. The pit is typically only filled with the original soil. The second aspect that makes the GES technology cost effective compared to traditional closed thermal storage systems is given by considering the dimensions needed to provide sufficient capacity to store energy in practice. The advantage of the open tank is that the heat energy is effectively stored by the earth in and below the tank, making its capacity extremely high in real application, providing in practice a multiple of the capacity that is making up the actual tank.

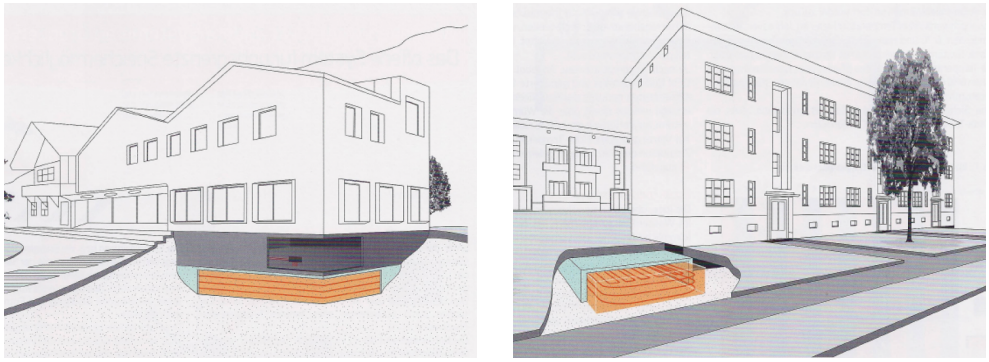


Figure 1. Real-world GES: under the new building (**left**) and during the renovation beside the building (**right**). Source: http://geo-ec.de/erdwaermetank/ (website access date: 25 May 2022).

We are especially interested in the potential of GES to store excess energy generated during the summer, e.g., by use of solar cells, for heating in winter. This becomes a very important issue when considering the entire heating system in dependence on weather conditions, day, night and annual rhythms over one year or even several years in time. Even though there are already working GES, particularly for single-family homes and smaller office buildings, well-founded evidence or simulations are required to assess the profitableness of a GES. This is of great importance with regard to the optimal dimensioning of the heat tank, which is the most expensive design factor. Moreover, this will also be very important in order to adopt this technique for large office complexes.

To study the reliability of such systems, it is important to know how they behave over long time spans for several months or years. The problem may be formulated in terms of a parabolic *partial differential equation* (PDE) model given by the heat equation related to space and time. In order to describe the heat transfer in the GES set-up adequately enough to tackle this task, the mathematical model including contact and boundary conditions must first be described. Subsequently, the simulation of heat evolution is run as long as the user requires. Let us stress in this context that the simulation may not be performed offline, since the planning of a GES may typically require communication between engineers, architects and the customer, often directly at construction sites. Thus, the simulation may ideally be performed at most in a few minutes during the discussion.

The simulation of heat transfer is of fundamental importance in diverse scientific fields; however, even nowadays it is still a challenging task to devise a method that combines reasonable accuracy and computational efficiency. Especially, when working with multi-dimensional problems, the computational performance of a solver is a key requirement. The long-term integration that needs to be performed in order to simulate seasonal energy storage represents an additional challenge. Standard methods are not designed to combine high efficiency and temporal accuracy, so a practical analysis of time integration methods for the heat equation in the context of GES is absolutely essential.

There exist countless methods to solve the heat equation PDE numerically. As examples, let us mention finite difference, finite volume or finite element methods, which

discretise the spatial dimensions into *ordinary differential equations* (ODEs) according to the method of lines procedure. Afterwards, the temporal integration can either be performed explicitly or implicitly. Explicit schemes are based on simple sparse matrix-vector multiplications in which the allowed time step size has a rather small upper bound, rendering the explicit strategy unsuitable for long-term evaluations. On the other hand, the use of implicit schemes leads to the task to solve a linear system of equations in each time step, whereby the number of variables related to the multi-dimensional GES may extend up to several million. The implicit schemes do not suffer from restrictions on the time step size τ in theory, but a fast solver for large sparse linear systems of equations is needed. To this end, sparse direct [1] or sparse iterative solvers [2] are commonly used. The direct solvers are characterised by computing very accurate solutions, but may be linked to high algorithmic complexity. In contrast, iterative methods have low complexity; however, they contain parameters that make their application not a straight forward task in a challenging setting. As another numerical aspect that has to be kept in mind when using implicit methods for applications involving source terms, one has to update the contributions of the sources using relatively small time intervals for obtaining an accurate simulation.

The classic solvers such as explicit schemes, sparse direct or sparse iterative solvers, are relatively simple to implement or based on existing sophisticated software packages; however, as indicated, their efficiency is related to severe time step size restrictions in the case of explicit methods, or high effort for solving large linear systems of equations when regarding implicit methods. For making any of these approaches efficient enough to tackle our application, two widely used techniques in their respective scientific fields called *fast explicit diffusion* (FED) and *model order reduction* (MOR), can be helpful to significantly reduce the computational effort compared to conventional methods.

The FED approach [3] originating from image processing, combines the advantage of an explicit evaluation with the possibility to achieve high integration times in just a few steps of evaluation. The underlying key idea is the use of suitable cycles with varying time step sizes, relying on a unique mixture of small and very large step sizes. On this basis, FED is substantially more efficient than the classic explicit scheme and it is based simultaneously on cheap matrix-vector multiplications. In order to avoid some drawbacks of the FED method, the advanced *fast semi-iterative* (FSI) method [4] can be used and is applied here; however, the design of the cycles requires special consideration.

The MOR methods represent another possible approach for reducing the computational complexity of PDE simulations. Such techniques can be applied to approximate the underlying ODE system (i.e., discretised in space but not in time at this stage) by a significantly reduced system, which is by its reduced dimension naturally much faster to solve. The idea of the MOR approach is that the reduced semi-discrete model preserves the main characteristics of the original ODE system. In the last decades, many different MOR methods have been developed, see, e.g., [5,6] for an overview; however, many of them are technically based on solving eigenvalue problems, Lyapunov equations or performing *singular value decomposition* (SVD), so that many approaches are limited by construction to the reduction in small- to medium-scale dynamical systems. For solving large-scale problems, the powerful *Krylov subspace model order reduction* (KSMOR) methods are most frequently used due to their superior numerical efficiency in this regime. This technique is directly related to matching the moments of the ODE system's transfer function. Common applications of KSMOR techniques are in the field of circuit simulation, power systems, micro-electro-mechanical systems and computational electromagnetics, e.g., also linked to parabolic PDEs [7–10]. Although the basic aspects of the KSMOR technique are well understood, it is not easy to devise it in a way that yields an efficient scheme for resolving heat evolution if internal boundary conditions with large input vectors are involved, as in our case.

Let us also mention that there exist other popular strategies to speed up diffusion processes. For instance, the alternating direction implicit method and its variations [11,12] or also operator splitting methods, such as additive operator splitting [13,14] or mul-

tiplicative operator splitting [13,15]. The main idea of these techniques is to split the multi-dimensional problem into several one-dimensional problems which can then be efficiently solved using tridiagonal matrix algorithms; however, this approach induces a splitting error that increases with the magnitude of the time step size τ. Moreover, external and internal boundary conditions have to be treated very carefully, which makes these methods difficult to apply in our setting.

1.1. Our Contributions

The long-term simulation of the heat equation related to the GES application requires advanced numerical methods specifically designed for the intended purpose. The main goal of this work is to discuss which methods are suitable for long-term simulations as appearing in real-world applications as considered here, and how they need to be used for the fundamental problem of heat evolution with internal and external boundary conditions as well as source terms. In this context, let us note that the methods we rely on are variations of schemes that already exist in the previous literature, and we show how to adapt them in order to obtain efficient schemes for the GES application. In some more detail, we provide the contributions below.

Considering our GES application, we precisely elaborate the complete continuous model and the corresponding discretisation, which is an important component for the numerical realisation. Especially, the matching conditions at the occurring interfaces lead to a large-dimensional input vector; therefore, they require special care when adapting the KSMOR technique. Since the long-term simulation of a three-dimensional GES is linked to extreme computational costs, it is a highly relevant point of interest if the model itself could be reduced for computational purposes. As one of our contributions, we demonstrate that the application of a two-dimensional and linear heat equation proves to be absolutely sufficient for the considered long-term simulation in our application. The latter is validated on real-world data by using temperature probes of a real, three-dimensional test field. The linear and dimensionally reduced model can then be used practically in place of the real (3D nonlinear) model, for either simulation purposes or parameter optimisation.

Let us turn to the numerical contributions of this work. We provide a comprehensive overview of the numerical solvers from various scientific areas that could be employed to tackle the intended task. In doing this, we thoroughly discuss the methods and the arising challenges in connection with long-term simulation of a GES. In order to improve the performance of the FED method, which has shown some promising first results [16] (comparing the implicit Euler method and the FED scheme for a simple synthesis experiment), the recent FSI scheme is employed. We show how to use the FSI method to parabolic problems including sources/sinks, relying on the discussion in [17]. Apart from that, we modify the original KSMOR technique for systems with a large number of inputs in a similar way as proposed in [9]. In this context, let us note that we provide here to our best knowledge the first very detailed exposition of this kind of approach. We validate our numerical findings at the hand of two experiments using synthetic and real-world data, and show that one can obtain fast and accurate long-term simulations of typical GES facilities.

In total, we give a detailed overview of relevant, modern numerical solvers that can be helpful for tackling long-term heat evolution in many engineering fields. For a complete insight into the methods regarding theoretical and numerical aspects, we refer the interested reader to [18].

1.2. Paper Organisation

This work is organised as follows. The Section 2 contains the continuous model description including modelling the external and internal boundary conditions, and we also inform about generating the initial heat distribution. Afterwards, we recall the numerical realisation by spatial and temporal discretisation in Section 3. In Section 4 an extensive overview of the numerical solvers is given, we discuss the methods FED, FSI, direct solvers, iterative solvers, KSMOR and KSMOR*. The experimental evaluation presented in Section 5

focuses on simulation quality and efficiency of the numerical solvers, by comparing two experiments. The paper is finished by a summary with our conclusion.

2. Continuous-Scale Mathematical Model

As indicated, the GES concept is especially designed for seasonal heat storage, so that a particular challenge is the long-term simulation. The underlying PDE for simulation of heat transfer is the heat equation which is a classic and thoroughly studied equation; however, a complete setup that covers a realistic scenario also includes heat sources and sinks, multiple boundary conditions and other physical properties such as the heat dissipation which may vary in space and time. The basic model equation that we consider in this manuscript is thus given by

$$\rho\, c\, \partial_t u(\boldsymbol{x}, t) = \operatorname{div}(\lambda \nabla u(\boldsymbol{x}, t)) + f(\boldsymbol{x}, t), \quad (\boldsymbol{x}, t) \in \Omega \times [0, t_F] \tag{1}$$

with thermal conductivity λ, density ρ, specific heat capacity c and heat source/sink f. In general, the physical quantities λ, ρ and c may vary in space and time and may even depend on the current temperature u in this equation. In addition, various boundary conditions including Dirichlet-, Neumann- and Robin boundary conditions will be needed. Indeed, the different regions depicted in Figure 2 consist of different media and thus, their heat transfer properties are different. This implies that interface boundary conditions are needed in our model.

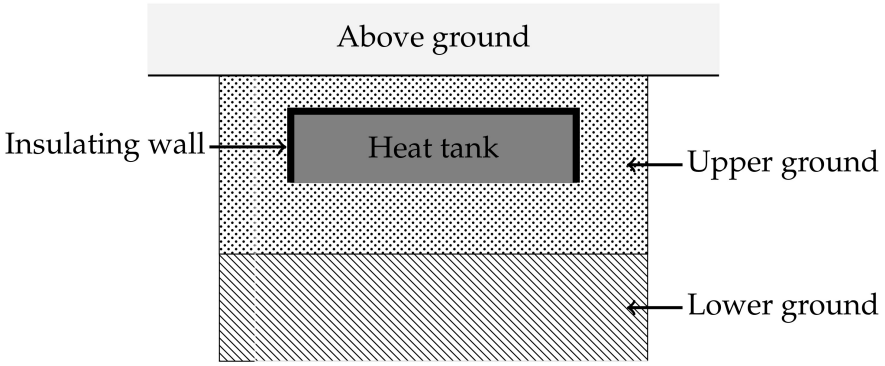


Figure 2. Cross-section as a schematic representation of a 3D-GES. The upper ground often has different properties than the lower ground. The heat tank is filled with upper ground, includes a heat source/sink and is downwardly open. Accordingly, the heat flows into the tank from below or inversely. Each interface exhibits different transition properties that need to be modelled into the heat flow equations.

2.1. Basic Model for Describing the Geothermal Energy Storage

As already mentioned, we aim to tackle the long-term evolution of heat in a GES as sketched in Figure 2. In the following, the physical variables λ, ρ and c are assumed to be constant and non-zero. These assumptions are reasonable in our near ground scenario even though the physical quantities are not perfectly constant, however, the temperature fluctuations are so small in the regime we are interested in, such that they may hardly cause any difference in the solutions of the PDE. In addition, we assume the surrounding soil to be non-porous and that seeping rainwater has no influence on the long-term evolution of the GES. The latter influence factor has only a short-term effect on the underground temperature, see [19].

Lastly, an important condition within the model of the GES is related to the groundwater, which may have a strong effect on the temperature distribution in the underground. In this work, we assume that the flowing groundwater has a large distance to the heat tank, so that one can specify isothermal boundary conditions for the lower ground boundary in our 3D model. The latter assumption facilitates not only the consideration of a pure heat

equation without a convection term, but based on this we may also neglect the percentage of water in the soil for specification of heat conductivity.

These assumptions ensure that the surrounding soils can be considered as homogeneous isotropic media. With this setup we define the thermal diffusivity $a := \frac{\lambda}{\rho c}$ and (1) can be rewritten into the following linear model PDE:

$$\partial_t u(\boldsymbol{x}, t) = a\Delta u(\boldsymbol{x}, t) + \frac{f(\boldsymbol{x}, t)}{\rho c}, \quad (\boldsymbol{x}, t) \in \Omega \times [0, t_F] \tag{2}$$

which has to be considered for each region sketched in Figure 2. The function f represents various simply modelled heat sources and sinks. Furthermore, an initial heat distribution $u(\boldsymbol{x}, 0)$ and boundary conditions (interface, border) are required as explained below.

We note that a more realistic source term model with heat exchanger pipes for the storage's charging and discharging is considered in the work [20], in which the authors focus in particular on the numerical simulation of the short-term behaviour of the GES.

2.2. Modelling of Interface Conditions

Concerning the interface conditions setup, we follow the general modelling framework as described, for instance, in [21]. The interface $\boldsymbol{x} = \boldsymbol{x}_i$ between two solids, where no heat is lost when considering the flux between them, can be modelled by means of the following boundary conditions:

$$\lambda^{(k)} \left(\partial_{\boldsymbol{n}} u^{(k)} \right) \Big|_{\boldsymbol{x}_i} = \lambda^{(l)} \left(\partial_{\boldsymbol{n}} u^{(l)} \right) \Big|_{\boldsymbol{x}_i} \tag{3}$$

Here, $u^{(k)}$ and $u^{(l)}$ are the solutions for the individual regions and where $\partial_{\boldsymbol{n}}$ denotes the derivative in outer normal direction.

Considering more specifically the contact between the upper ground and the lower ground, we assume that contact is perfect and that heat transfer is continuous, so that it additionally follows at the interface

$$u^{(k)}(\boldsymbol{x}_i, t) = u^{(l)}(\boldsymbol{x}_i, t) \tag{4}$$

In contrast, the interface involving the insulating walls is subjected to thermal contact resistance, which in general may lead to a discontinuity in heat transfer, and yields the condition

$$\lambda^{(k)} \left(\partial_{\boldsymbol{n}} u^{(k)} \right) \Big|_{\boldsymbol{x}_i} = \alpha_c \left(u^{(l)} - u^{(k)} \right), \quad \lambda^{(l)} \left(\partial_{\boldsymbol{n}} u^{(l)} \right) \Big|_{\boldsymbol{x}_i} = \alpha_c \left(u^{(l)} - u^{(k)} \right) \tag{5}$$

with a contact heat transfer coefficient $\alpha_c \geq 0$.

Consequently, the interaction between different types of soils at interface $\boldsymbol{x}_i$ can be modelled by Equations (3) and (4). Based on this, the interface between soil and insulation is modelled via (3) and (5).

2.3. Modelling of Boundary Conditions

External boundary conditions must also be specified within the model, compare Figure 2; therefore, we have to fix conditions situated on $\partial\Omega$, actually at the top, at the bottom and at the sides.

The upper domain boundaries (i.e., between upper and above ground, cf. Figure 2) are characterised such that the soil of the upper ground is not under roof of other structures. Thus, time-dependent Robin boundary conditions

$$-\lambda^{(k)} \left(\partial_{\boldsymbol{n}} u^{(k)} \right) \Big|_{\boldsymbol{x}_i} = \alpha_A \left(u^{(k)} - T_A(t) \right) \tag{6}$$

are considered at the interface between the topmost layer of ground and the air above, where $T_A(t)$ is the ambient temperature on the earth surface and α_A the heat transfer

coefficient, see [21]. The coefficient α_A can either be assumed constant or dependent on current weather conditions.

The lateral domain boundaries on $\partial\Omega$ are generally unknown and may be affected by various factors, such as for instance border-near basements. Ideally, we assume that there exist no anthropogenic influences; therefore, we specify within the model time-dependent Dirichlet boundary conditions in the form of undisturbed ground temperatures $T_g(t,x)$, which also depend on space (i.e., on depth in the ground, in meters) in the following form:

$$T_g(t,x) = \bar{\theta} - \delta\theta \exp\left(-\frac{x}{\delta_g}\right) \cos\left[2\pi\frac{(t-\bar{t})}{t_h} - \frac{x}{\delta_g}\right] + G_t x \tag{7}$$

with average ambient temperature $\bar{\theta}$, amplitude of monthly fluctuations in ambient temperature $\delta\theta$, measure of lagging ambient temperature in depth δ_g, number of hours in year t_h, time lag between the time of the lowest annual temperature $\bar{t}$ and geothermal gradient G_t. Thereby, the value δ_g is given by

$$\delta_g = \sqrt{\frac{3600 t_h \lambda}{\pi \rho c}} \tag{8}$$

In particular, the lateral boundary condition (7) depends on the location of the installed GES due to the location-dependent parameters $\bar{\theta}, \delta\theta, \delta_g$ and G_t.

Finally, the lower domain boundaries are given by time-independent and constant Dirichlet boundary conditions in form of groundwater temperatures

$$T_{gw}(t) = \widetilde{g} \tag{9}$$

2.4. Generating the Initial Heat Distribution

The initial condition of the GES model problem, which corresponds to the initial temperatures at time $t = 0$, is generally not known. In practice, initial temperatures may only be determined by mounting temperature sensors $\widetilde{s}$ at some grid points. In the presence of sensor data, the initial temperature is known in some places $\boldsymbol{x}_j \in \Omega_K \subset \Omega$, but most of the initial heat distribution $u(\boldsymbol{x}, 0)$ remains unknown. One possibility is to estimate the unknown temperatures by interpolation using the given data. In doing so, the interpolation task can be realised by PDE-based *image inpainting*, also known as Laplace interpolation, see, e.g., [22,23]. More precisely, image inpainting is a process in order to reconstruct or fill-in missing parts in the inpainting domain $\Omega \backslash \Omega_K$ in a way that is undetectable to the casual observer.

To this end, in turn, the heat equation can be used. In the simplest case of applying linear diffusion, the *initial heat distribution* is obtained as the steady state solution, i.e.,

$$u(\boldsymbol{x}, 0) := \lim_{t \to \infty} \widetilde{u}(\boldsymbol{x}, t) \tag{10}$$

of the heat evolution that is described by

$$\begin{aligned} \partial_t \widetilde{u}(\boldsymbol{x}, t) &= \Delta \widetilde{u}(\boldsymbol{x}, t), && \forall \boldsymbol{x} \in \Omega \backslash \Omega_K \\ \widetilde{u}(\boldsymbol{x}, 0) &= 0, && \forall \boldsymbol{x} \in \Omega \backslash \Omega_K \\ \widetilde{u}(\boldsymbol{x}, t) &= \widetilde{s}(\boldsymbol{x}), && \forall \boldsymbol{x} \in \Omega_K \end{aligned} \tag{11}$$

with suitable boundary conditions on $\partial\Omega$. In other words, the temperature sensors on Ω_K are interpreted as thermostats that are kept at a fixed temperature at any time, and are thus modelled as Dirichlet boundary conditions.

2.5. Summary on Modelling

For the correct modelling of the GES application, various boundary conditions must be carefully treated. We briefly summarise the mathematical modelling in Table 1 below.

Table 1. Summary of the GES model presented in Section 2 regarding boundary conditions and their specific region, type as well as the corresponding equation.

Boundary Conditions	Region	Type	Equation
External	Top	Time-dependent Robin	(6)
	Bottom	Time-dependent Dirichlet	(7)
	Lateral	Time-independent Dirichlet	(9)
Internal	Soil–Soil	Continuous heat transfer	(3), (4)
	Soil–Insulation	Discontinuous heat transfer	(3), (5)
Initial	Complete	Laplace interpolation	(10)

3. Discretisation of the Continuous-Scale Model

In this section, we provide the basic discretisation of the underlying continuous GES model, characterised via linear heat Equation (2), different interior (3)–(5) and exterior (6)–(7), (9) boundary conditions and initial heat distribution (10). In doing so, we describe the discretisation aspects in space and time in detail in the following subsections.

The underlying computational domain (cf. Figure 2) is given by a cuboid type form, so that we apply standard finite difference schemes for the discretisation of the continuous model, which will be sufficient for the application in this work.

3.1. Discretisation in Space

For reasons of simplicity, we consider in the following the two-dimensional rectangular domain $(x, y) = [x_1, x_n] \times [y_1, y_m] \in \Omega$ with an equidistant mesh size $h = \Delta x = \Delta y > 0$ in x- and y-direction, where $u_{i,j}(t)$ denotes an approximation of the unknown function u at grid point (x_i, y_j) and time t. For convenience only, we use the abbreviation $u_{i,j} := u_{i,j}(t)$.

The approximation of the spatial partial derivatives u_{xx} and u_{yy} in (2) using standard central differences leads to

$$\frac{\mathrm{d}u_{i,j}}{\mathrm{d}t} = a\frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2} + \frac{f_{i,j}}{\rho c} \tag{12}$$

where $f_{i,j}$ is the discretised heat source/sink.

As mentioned before, the interaction between different types of soils is modelled by Equations (3) and (4). In particular, we assume that no grid point is located exactly at the interface $\boldsymbol{x}_i$. Exemplary, we define the interface as $\boldsymbol{x}_i := (x_{i+\frac{1}{2}}, y_j)$, which is centrally located between the grid points (x_i, y_j) and (x_{i+1}, y_j). Discretising (3) at the interface $\boldsymbol{x}_i$ between two layers, denoted here as "k" and "l", using the forward and backward difference results in

$$\lambda^{(k)}\frac{u^{(k)}_{i+1,j} - u^{(k)}_{i+\frac{1}{2},j}}{\frac{h}{2}} = \lambda^{(l)}\frac{u^{(l)}_{i+\frac{1}{2},j} - u^{(l)}_{i,j}}{\frac{h}{2}} \tag{13}$$

Due to (4), we have $u^{(k)}_{i+\frac{1}{2},j} = u^{(l)}_{i+\frac{1}{2},j} =: u_I$ for the fictitious value u_I at the interface, which can then be calculated via

$$u_I = \frac{\lambda^{(k)}u^{(k)}_{i+1,j} + \lambda^{(l)}u^{(l)}_{i,j}}{\lambda^{(k)} + \lambda^{(l)}} \tag{14}$$

The latter scheme is visualised on the left in Figure 3. Let us mention that the discretisation can also be achieved without using a fictitious value if one assumes that a grid point lies on the interface, see [24].

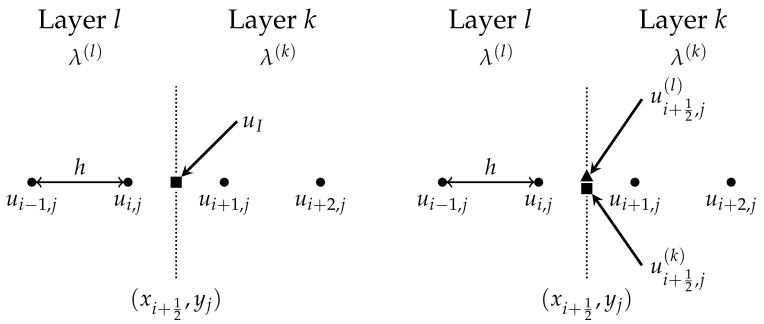


Figure 3. Schematic sketch at the interface $\boldsymbol{x}_i = (x_{i+\frac{1}{2}}, y_j)$ between layers k and l using the proposed finite difference scheme at grid points (dots) and equidistant mesh size h. **Left:** Discretisation of the interaction between different types of soils leads to fulfilling the matching conditions (3) and (4) around the fictitious interface point, and defining the value u_I using the Formula (14). **Right:** Discretisation of the interaction between soil and insulation leads to fulfilling the matching conditions (3) and (5) around the fictitious interface points, and defining left and right values $u^{(k)}_{i+\frac{1}{2},j}$ and $u^{(l)}_{i+\frac{1}{2},j}$ using the system of linear Equation (18).

In contrast, modelling the relation between soil and insulation, condition (4) must be replaced by (5). Discretisation of (5) at $\boldsymbol{x}_i$ using the forward and backward differences gives

$$\lambda^{(k)} \frac{u^{(k)}_{i+1,j} - u^{(k)}_{i+\frac{1}{2},j}}{\frac{h}{2}} = \alpha_c \left(u^{(l)}_{i+\frac{1}{2},j} - u^{(k)}_{i+\frac{1}{2},j} \right) \tag{15}$$

which can be rewritten as

$$\left(\frac{2\lambda^{(k)}}{h} - \alpha_c \right) u^{(k)}_{i+\frac{1}{2},j} + \alpha_c u^{(l)}_{i+\frac{1}{2},j} = \frac{2\lambda^{(k)}}{h} u^{(k)}_{i+1,j} \tag{16}$$

Moreover, the Equation (13) can be transformed into

$$u^{(k)}_{i+\frac{1}{2},j} + \frac{\lambda^{(l)}}{\lambda^{(k)}} u^{(l)}_{i+\frac{1}{2},j} = u^{(k)}_{i+1,j} + \frac{\lambda^{(l)}}{\lambda^{(k)}} u^{(l)}_{i,j} \tag{17}$$

Equations (16) and (17) form a system of linear equations for the two unknowns $u^{(k)}_{i+\frac{1}{2},j}$ and $u^{(l)}_{i+\frac{1}{2},j}$ by means of

$$\begin{pmatrix} 1 & \frac{\lambda^{(l)}}{\lambda^{(k)}} \\ \frac{2\lambda^{(k)}}{h} - \alpha_c & \alpha_c \end{pmatrix} \begin{pmatrix} u^{(k)}_{i+\frac{1}{2},j} \\ u^{(l)}_{i+\frac{1}{2},j} \end{pmatrix} = \begin{pmatrix} u^{(k)}_{i+1,j} + \frac{\lambda^{(l)}}{\lambda^{(k)}} u^{(l)}_{i,j} \\ \frac{2\lambda^{(k)}}{h} u^{(k)}_{i+1,j} \end{pmatrix} \tag{18}$$

The latter system has a unique solution for $\alpha_c \neq \frac{2\lambda^{(k)}\lambda^{(l)}}{h(\lambda^{(k)}+\lambda^{(l)})}$ and can be solved with Cramer's rule. The used scheme at the interface including a jump condition is shown on the right in Figure 3. We mention that another possible discretisation is presented in [24].

Lastly, the exterior boundary conditions must be discretised. In case of the upper domain boundaries at the topmost layer of ground, interfaces $\boldsymbol{x}_i = (x_i, y_m)$ and fictitious values $u^{(k)}_{i,m+1}$ for $i = 1, \ldots, n$ are again incorporated. Using the standard first-order spatial discretisation

$$\left(\partial_y u^{(k)} \right) \Big|_{(x_i, y_m)} = \frac{u^{(k)}_{i,m+1} - u^{(k)}_{i,m}}{h} + \mathcal{O}(h) \tag{19}$$

linked to the discretised condition (6) as

$$-\lambda^{(k)}\left(\partial_y u^{(k)}\right)\Big|_{(x_i,y_m)} = \alpha_A\left(u_{i,m}^{(k)} - T_A(t)\right) \tag{20}$$

leads to

$$u_{i,m+1}^{(k)} = u_{i,m}^{(k)} - \frac{h\alpha_A}{\lambda^{(k)}}\left(u_{i,m}^{(k)} - T_A(t)\right) \tag{21}$$

Let us note that, when using first-order discretisations, the accuracy at the boundary is formally by one order worse than the error of the boundary-free discrete equation; however, it is a matter of definition of boundary location to understand the same finite difference expression as a central discretisation of the derivative between the considered points, which is again of second order—this means we do not have to expect any error deterioration. What is important here is the symmetry preservation of the underlying Laplacian matrix L, which, for later application of the numerical solvers, is of great importance.

The remaining conditions concerning the lateral and lower domain boundaries are fixed via

$$u_{0,j}^{(k)} = u_{n+1,j}^{(k)} = T_g(t, d_j) = T_{g_j}, \quad j = 1, \ldots, m \tag{22}$$

$$u_{i,0}^{(k)} = \widetilde{g}, \quad i = 1, \ldots, n \tag{23}$$

where d_j is the depth of the j-th grid layer. Let us mention that the three-dimensional case can be handled analogously.

3.2. Arising System of Ordinary Differential Equations

Let us now summarise the components of the proposed two-dimensional discretisation (12), (14), (18) and (21)–(23) which end up in a semi-discrete ODE system. In particular, a function defined on all grid points may be represented by now as an N-dimensional vector

$$\boldsymbol{u}(t) = \left(u_1(t), \ldots, u_N(t)\right)^\top \tag{24}$$

where N is the total number of all grid points with linear grid point numbering from top left to bottom right.

The proposed spatial discretisation of the GES by applying finite differences on a regular grid with constant grid size h results into an ODE system, one ODE for each grid point, with the notation $\dot{\boldsymbol{u}} = \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}$ as follows:

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K_1\boldsymbol{u}_1(t) + K_2\boldsymbol{u}_2(t) + K_A T_A(t) + K_{gw}\widetilde{g} + K_g\boldsymbol{T}_g(t) + K_f\boldsymbol{f}(t) \tag{25}$$

with temperature vector $\boldsymbol{u} \in \mathbb{R}^N$, temperature vectors $\boldsymbol{u}_1 \in \mathbb{R}^n$ (continuity condition) and $\boldsymbol{u}_2 \in \mathbb{R}^{\widetilde{n}}$ (discontinuity condition) for fictitious points at the two different types of interfaces, ambient temperature $T_A \in \mathbb{R}$, groundwater temperature $\widetilde{g} \in \mathbb{R}$, undisturbed ground temperature vector

$$\boldsymbol{T}_g(t) = \left(T_{g_1}(t), \ldots, T_{g_m}(t)\right)^\top \in \mathbb{R}^m \tag{26}$$

and source/sink vector $\boldsymbol{f} \in \mathbb{R}^{\widetilde{m}}$. The conditions at the lateral boundaries on Ω considered here are identical, therefore it is sufficient that the input vector $\boldsymbol{T}_g$ is of size m. At this point it should be mentioned that the values for $\widetilde{n}$ and $\widetilde{m}$ depend on the user-defined size setting of the insulating walls and geometry setting of the considered source/sink, respectively.

The matrices $L \in \mathbb{R}^{N\times N}$, $K_1 \in \mathbb{R}^{N\times n}$, $K_2 \in \mathbb{R}^{N\times\widetilde{n}}$ and $K_f \in \mathbb{R}^{N\times\widetilde{m}}$ collect the terms of the basic discretisation and are not shown in detail. The remaining matrices, which

correspond to the ambient temperature, the groundwater temperature and the undisturbed ground temperature, have the following simple structure:

$$K_A = \frac{\alpha_A}{\rho c h}\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ \hline 1 \\ 0 \\ \vdots \\ 0 \\ \hline \vdots \end{pmatrix}, \quad K_{gw} = \frac{a}{h^2}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \hline 1 \\ 0 \\ \vdots \\ 0 \\ \hline 1 \\ \vdots \end{pmatrix}, \quad K_g = \frac{a}{h^2}\begin{pmatrix} I_{m,m} \\ \hline 0_{N-2m,m} \\ \hline I_{m,m} \end{pmatrix} \tag{27}$$

with $K_A, K_{gw} \in \mathbb{R}^N$, $K_g \in \mathbb{R}^{N\times m}$, the identity matrix $I_{m,m} \in \mathbb{R}^{m\times m}$. the null matrix $0_{N-2m,m} \in \mathbb{R}^{(N-2m)\times m}$, and where a, ρ, c depend on material parameters. The mark "—" within (27) indicates that the discretisation points of the underlying rectangular computational domain are considered row by row. Finally, the discrete initial condition is given by (10) with $\boldsymbol{u}^0 := \boldsymbol{u}(\boldsymbol{x}, 0)$.

3.3. Time Integration

The application of the proposed spatial discretisations leads to the ODE system (25), which can be represented as

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K\boldsymbol{w}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{u}^0,\ \boldsymbol{w}(0) = \boldsymbol{w}^0 \tag{28}$$

with input matrix and input vector

$$K = \left[K_1, K_2, K_A, K_{gw}, K_g, K_f\right], \quad \boldsymbol{w}(t) = \left(\boldsymbol{u}_1(t), \boldsymbol{u}_2(t), T_A(t), \widetilde{g}, \boldsymbol{T}_g(t), \boldsymbol{f}(t)\right)^\top \tag{29}$$

Thereby, we specify the input matrix $K \in \mathbb{R}^{N\times p}$ with $p = n + \widetilde{n} + 2 + m + \widetilde{m}$ and make use of stacked vectors for defining $\boldsymbol{w}(t) \in \mathbb{R}^p$. Let us comment that the time-dependent large-scale input $\boldsymbol{w}(t)$ controls the model by boundary conditions and heat sources/sinks. Further we note, although the matching conditions at the interfaces corresponding to $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are not user-defined controls, one has to handle these as indirect inputs. In addition, the Laplacian matrix L is symmetric and negative definite which is large, sparse and structured. The definiteness follows directly from its strictly diagonal dominance and negative diagonal entries according to Gershgorin's circle theorem [25].

The most frequently used class for solving a system of ODEs are time stepping methods, in which the time variable in (28) is discretised by $0 = t_0 < t_1 < \cdots < t_J = t_F$. Discrete time stepping methods of ODEs can be achieved using standard numerical integration so-called *time integration methods*; for an excellent overview, we refer to [26,27]. Let us briefly recall the standard methods that we apply in this work. Common time integration schemes are the *explicit Euler* (EE) method, the *implicit Euler* (IE) method and the trapezoidal rule known as *Crank–Nicolson* (CN) method. Usually, the implicit CN method is preferably used compared to IE due to its second-order convergence in time and at the same time marginally higher computational costs. Consequently, besides the EE method, we consider here only the CN method for the numerical solution of the underlying model problem. Let us note that the IE method has been investigated in our previous conference paper [16].

To apply time discretisation methods, time intervals $I_k = [t_k, t_{k+1}]$ are defined in order to subdivide the complete integration time $[0, t_F]$ into a partition. The resulting numerical methods then generate approximations $\boldsymbol{u}(t_k)$ at the different time levels t_k. We mention that uniform partitions of time intervals are used in this work.

3.3.1. Explicit Euler

The use of the fundamental lemma of calculus for the left-hand side and the left-hand rectangle method for the approximation of the integral on the right-hand side of (28) over the time interval I_k gives

$$\int_{t_k}^{t_{k+1}} L\boldsymbol{u}(t) + K\boldsymbol{w}(t)\,\mathrm{d}t \approx \tau L\boldsymbol{u}(t_k) + \tau K\boldsymbol{w}(t_k) \tag{30}$$

with the uniform time step size $\tau = t_{k+1} - t_k$. Finally, using the notation $\boldsymbol{u}(t_k) = \boldsymbol{u}^k$, $\boldsymbol{w}(t_k) = \boldsymbol{w}^k$, the well-known fully discrete EE method

$$\boldsymbol{u}^{k+1} = (I + \tau L)\boldsymbol{u}^k + \tau K\boldsymbol{w}^k \tag{31}$$

with $k \in \{0, \ldots, J-1\}$, the identity matrix $I \in \mathbb{R}^{N\times N}$ and the given data $\boldsymbol{u}^0 = \boldsymbol{u}(0)$, $\boldsymbol{w}^0 = \boldsymbol{w}(0)$ is obtained. Due to the fact that the values $\boldsymbol{u}^k$ and $\boldsymbol{w}^k$ at time t_k are known, the new values $\boldsymbol{u}^{k+1}$ at time t_{k+1} can easily be computed by simple sparse matrix-vector multiplication. Schemes in this form are known as *explicit methods* and are well-suited for parallel computing such as *graphics processing units* (GPUs); however, it is well-known that explicit methods such as EE are only conditionally stable, see, e.g., [28] and the stability requirement leads to a severe limitation in the size of the time step τ. In general, the typical time step size restriction has a rather small upper bound in the case of stiff ODE systems, more precisely the stability requirement, which depends quadratically on the spatial grid size, yields a time step size limitation $\tau_{\max}$. As a consequence, explicit schemes are usually considered to be extremely inefficient numerical methods from a computational point of view.

3.3.2. Crank–Nicolson

Moreover, aside the limitation on the stability requirement on τ, the EE method is additionally characterised just by first-order approximation to the exact solution. A common alternative is the implicit CN method, which is second-order in time and moreover numerically stable. Applying the trapezoidal rule for the integral approximation of the right-hand side of (28) gives

$$\int_{t_k}^{t_{k+1}} L\boldsymbol{u}(t) + K\boldsymbol{w}(t)\,\mathrm{d}t \approx \frac{\tau}{2}\big(L\boldsymbol{u}(t_{k+1}) + L\boldsymbol{u}(t_k) + K\boldsymbol{w}(t_{k+1}) + K\boldsymbol{w}(t_k)\big) \tag{32}$$

which leads to the popular fully discrete CN scheme

$$\big(I - \tfrac{\tau}{2}L\big)\boldsymbol{u}^{k+1} = \big(I + \tfrac{\tau}{2}L\big)\boldsymbol{u}^k + \tfrac{\tau}{2}K\Big(\boldsymbol{w}^{k+1} + \boldsymbol{w}^k\Big) \tag{33}$$

Computing the values $\boldsymbol{u}^{k+1}$ at time t_{k+1} requires the solution of a system of linear equations as well as a sparse matrix-vector multiplication in each time step. Consequently, this scheme is more numerically intensive than the EE method; however, the considerable advantage of an implicit method is the numerical stability independently of the time step size τ, cf. [28]. At this point, it should be noted that the CN method is sensitive to problems with discontinuous initial conditions and can lead to undesirable oscillations in the numerical solution, but this is not the case with the GES application.

3.4. Summary on Discretisation

To solve the GES model problem numerically, the continuous PDE model must be discretised. Using the method of lines, i.e., discretising the spatial derivatives first, a semi-discretised system of ODEs (28) with only one independent variable is obtained. Afterwards an approximate solution for the initial value ODE problem must be computed; therefore, a temporal grid is introduced. Time integration methods are typically used to design a nu-

merical scheme. Here, we recalled the first-order EE scheme (31) and the second-order CN method (33). These baseline schemes are mainly used in our work for comparison purposes.

4. Numerical Solvers

As seen in the last section, the temporal integration can either be achieved explicitly or implicitly. For both approaches, there exist several numerical solvers, which have different advantages in terms of computational effort and accuracy of the computed solution.

Classic explicit schemes require low computational effort when resolving one time step, but they are just conditionally stable and may suffer from small time step size restrictions. In order to improve the performance, we have shown in our conference paper [16] the useful application of FED. To overcome drawbacks of this method, we highlight some relevant properties of the FSI method that have not been discussed in this context earlier. Furthermore, we introduce for the first time KSMOR methods in connection with the GES application. Due to large input vectors (29) within the semi-discrete GES model the use of KSMOR leads to new challenges. For this reason, we follow a recent approach [9] to enhance the efficiency of the original technique. Finally, the viability of the modified KSMOR method is demonstrated using a test example.

In the following, we provide a comprehensive overview of numerical solvers for solving the underlying GES model problem, for a deeper insight see [18].

4.1. Explicit Methods

Explicit methods are simple iterative schemes of the form such as (31); however, in general, the time step size restriction $\tau_{\max}$ has a rather small upper stability bound and makes these methods unsuitable for long-term simulations. An alternative is the use of *fast explicit methods*, namely FED or FSI, which are well-known in image processing.

Let us mention that the knowledge of the value of $\tau_{\max}$ is necessary for using FED as well as FSI; however, this poses no problem because the step size $\tau_{\max}$ is limited with respect to the largest modulus of the eigenvalues of L; therefore, using Gershgorin's circle theorem, it is possible to obtain an estimate of the upper bound for $\tau_{\max}$ to ensure stability.

4.1.1. Fast Explicit and Fast Semi-Iterative Diffusion

The FED and FSI methods are recent schemes from image processing. For a detailed presentation, we refer to the manuscripts [3,4,18,29] and the excellent monograph [17]. The core idea behind FED (see Appendix A.1 for a brief insight) is to consider an EE scheme and interleave time steps that significantly violate the upper stability bound $\tau_{\max}$ with small stabilising steps. The idea is related to super time stepping methods as presented much earlier in [30]. Overall, the FED method benefits from its explicit scheme structure and is therefore also well-suited for modern parallel architectures such as GPUs, see [29,31]. In addition, this method is also appealing because its implementation causes almost no overhead compared to the classic EE scheme.

Nevertheless, the FED scheme has two major drawbacks from a numerical point of view. Although the internal stability of the scheme is theoretically fulfilled, FED is highly sensitive to numerical rounding errors. Thus, a rearrangement of varying time step sizes is required in order to avoid serious accumulation of rounding errors. As a consequence, such a rearrangement can yield highly unstable intermediate solutions, and therefore the input $\boldsymbol{w}^k$ should be kept constant within a FED-cycle (cf. (A2)) and is only updated after a full cycle. For this reason, the well-performing FSI scheme is better suited for solving GES.

Fast Semi-Iterative Diffusion

In order to reduce numerical rounding errors and simultaneously to increase the approximation quality, FSI uses a semi-iterative approach where the new value $\boldsymbol{u}^{k+1}$ is computed via $\boldsymbol{u}^k$ and $\boldsymbol{u}^{k-1}$. For linear problems $L(\boldsymbol{u}^k) = L$ and time-independent inputs $\boldsymbol{w}(t) = \boldsymbol{w}$, one can even show that FED and FSI yield identical results. In contrast to FED, the FSI scheme only uses stable time step sizes and ensures internal stability so that strong

numerical rounding errors can be avoided. As a result, no rearrangements of time step sizes are necessary and updates of the input vector $\boldsymbol{w}^k$ within one cycle are allowed. Based on these properties, the proposed FSI scheme is highly beneficial.

In total, the cyclic FSI scheme, more detailed described in works [4,17,18], for the m-th cycle with cycle length n is given by

$$\boldsymbol{u}^{m,k+1} = \alpha_k \Big[(I + \tau L)\boldsymbol{u}^{m,k} + \tau K \boldsymbol{w}^{m,k} \Big] + (1 - \alpha_k)\boldsymbol{u}^{m,k-1} \quad (k = 0, \ldots, n-1)$$

with

$$\begin{aligned} &\boldsymbol{u}^{m,-1} := \boldsymbol{u}^{m,0}, \quad n = \left\lceil \sqrt{\frac{3t_F}{\tau_{\max} M} + \frac{1}{4}} - \frac{1}{2} \right\rceil, \quad \alpha_k = \frac{4k+2}{2k+3}, \quad \tau = \frac{3t_F}{Mn(n+1)} \\ &\boldsymbol{w}^{m,k} := \boldsymbol{w}^m(c_k), \quad c_{k+1} = \alpha_k(c_k + \tau) + (1 - \alpha_k)c_{k-1}, \quad c_0 = c_{-1} = 0 \end{aligned} \tag{34}$$

where M is the number of outer FSI cycles, t_F the stopping time, $\tau_{\max}$ the theoretical upper bound for a stable EE scheme and $\boldsymbol{w}^{m,k}$ is the input at the increment parameter c_k of the m-th cycle. Increasing the number of cycles M, whereby n becomes smaller, improves the approximation quality, but is directly connected with higher computational costs.

Although FSI requires an additional computation of the sum of two vectors, the computational costs are less than FED because non-varying time step sizes are used within the matrix-vector multiplications in (34), meaning $I + \tau L$ must be computed only once. Another benefit of FSI is its straightforward use for nonlinear problems $L(\boldsymbol{u}^k)$, where the scheme allows us to perform nonlinear updates within one cycle.

The accelerated explicit methods FED and FSI can be applied for parabolic problems whenever the matrix L is symmetric. It can then be shown that the iterates $\boldsymbol{u}^{k+1}$ are stable in the l^2 sense. Furthermore, we emphasise that a constant step size $\tau < \tau_{\max}$ often yields stability in the l_∞ sense.

Let us also note that FED and FSI are mainly applied to problems in image processing or computer vision, see, e.g., [4,32,33]. However, the schemes can be applied to many parabolic problems, including time-dependent boundary condition, also in an engineering context as demonstrated in this work. Beyond this, FSI can also be translated to neural architectures and thus offers a high level of practical relevance, we refer the interested reader to the current work [34].

We point out that the mentioned explicit generalisations such as FED and FSI have a natural weakness. Both methods perform inefficiently for highly non-uniform meshes, especially when also very small grid widths arise, compare [35]. This is based on the fact that the (minimum) spatial mesh width and the allowed time step size of explicit methods are coupled. In this work, however, it is sufficient to consider (relatively) uniform grids in the context of the GES application.

4.2. Implicit Methods

As mentioned in Section 3.3, the CN method is an implicit scheme and results in a system of linear equations. More precisely, (33) can be expressed as a linear system

$$A\boldsymbol{x} = \boldsymbol{b} \tag{35}$$

using the notation $A = I - \frac{\tau}{2}L$ and $\boldsymbol{b} = \left(I + \frac{\tau}{2}L\right)\boldsymbol{u}^k + \frac{\tau}{2}K\Big(\boldsymbol{w}^{k+1} + \boldsymbol{w}^k\Big)$. The underlying constant matrix $A \in \mathbb{R}^{N \times N}$ is symmetric, positive definite and also large, sparse and structured. Implicit methods are unconditionally stable independently of the time step size τ, but solving linear equations requires significant computational effort and therefore a fast solver for large sparse systems of linear equations is necessary. Moreover, the choice of the time step size τ has to take into account since an increase in τ leads to a less accurate solution, see, e.g., [14]. The linear system (35) can be solved with *sparse direct* and *sparse iterative* solvers.

4.2.1. Sparse Direct Solvers

The application of sparse direct solvers, generally related to Gaussian elimination and *lower-upper* (LU) factorisation, compute highly accurate solutions (except of numerical rounding errors) and are predestined for solving a linear system with multiple right-hand sides. In that case, the underlying symmetric matrix A will be factorised (Cholesky factorisation) just one time, and subsequently for each right-hand side, the system is solved by forward and backward substitution; however, a fast computation requires a complete factorisation, which may use substantial memory. Another drawback in this context is that the factorisation is harder to optimise, since the factors usually do not preserve the sparsity, and to design a parallelised version. Consequently, the memory and runtime depends on the size of the underlying system. We refer the reader to [1,36] and the reference therein for more details on sparse direct methods.

4.2.2. Sparse Iterative Solvers

The general idea of iterative methods is to successively compute a sequence of approximate solutions $\{\boldsymbol{x}_k\}$, starting from a given initial $\boldsymbol{x}_0$, which converges to $\boldsymbol{x}^\star = A^{-1}\boldsymbol{b}$, i.e.

$$\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \overset{k\to\infty}{\longrightarrow} \boldsymbol{x}^\star \in \mathbb{R}^N : A\boldsymbol{x}^\star = \boldsymbol{b} \tag{36}$$

In particular, sparse iterative solvers are less accurate compared to direct methods but are very fast in computing approximate solutions. These are built on sparse matrix-vector multiplication and are therefore also easier to optimise and parallelise. They require much less memory space and are thus inherently attractive candidates for multi-dimensional simulations. Although iterative solvers are characterised by low algorithmic complexity, the convergence of the iteration procedure is not guaranteed. Overall, the runtime of iterative methods depends on the data, size, sparsity and required accuracy and makes these methods a tool that is not straightforward to use. For a more detailed insight into iterative methods, we refer to [2,37,38].

Conjugated Gradient Method

We propose to employ the well-known *conjugate gradient* (CG) method of Hestenes and Stiefel [39] (see Appendix A.2 for a brief insight), which is still an adequate iterative solver for problems involving large sparse symmetric and positive definite matrices. For the CG method, one can show that the approximate solutions $\boldsymbol{x}_l$ (in the l-th iteration) are optimal in the sense that they minimise the so-called energy norm of the error vector [40]. In other words, the CG method gives in the l-th iteration the best solution available in the generated subspace. Without computational rounding errors, the theoretical convergence will be achieved at latest after the N-th step of the method if the sought solution is in $\mathbb{R}^N$.

Applying the CG method practically requires a stopping criterion. Typically, the *relative residual* $\frac{\|\boldsymbol{r}_k\|_2}{\|\boldsymbol{b}\|_2} = \frac{\|\boldsymbol{b}-A\boldsymbol{x}_k\|_2}{\|\boldsymbol{b}\|_2} \leq \varepsilon$ is used in practice. Increasing ε leads naturally to faster computations but somewhat worse results. Usually, $\varepsilon = 10^{-6}$ is used, nevertheless the selected value will depend on the problem being solved. Previous works [16,41] built upon the discrete Laplacian operator validated a favourable acceptance range of $\varepsilon \in [10^{-5}, 10^{-3}]$.

Although the CG method relates to ideal theoretical properties, in practice, numerical rounding errors appear and one may suffer from convergence problems for very large systems. Thus, a *preconditioning* is recommended to enforce all the beneficial properties of the algorithm, along with fast convergence.

Preconditioned Conjugate Gradient Method

In simple terms, preconditioning is used to reduce the condition number of the system matrix and consequently improve the performance of the Krylov solver. The combination of CG and the preconditioning technique is known as the *preconditioned conjugate gradient* (PCG) method. The basic idea of preconditioning is to multiply the original system $A\boldsymbol{x} = \boldsymbol{b}$ from the left with a matrix P that is close to A^{-1}. The modified system $PA\boldsymbol{x} = P\boldsymbol{b}$ is

generally better conditioned (small condition number close to one) and is much more efficient to solve; however, PA is typically not sparse, symmetric and positive definite, so the CG method cannot be applied; therefore, a *two-sided* preconditioner preconditioning strategy must be chosen.

The main challenge is to construct a suitable preconditioner P with a good cost balance for the computation of the preconditioner itself and the resulting efficiency per PCG iteration. Dealing with parabolic PDEs and underlying symmetric matrices, the *incomplete Cholesky* (IC) decomposition and its variation *modified incomplete Cholesky* (MIC) decomposition is often used for constructing a common and very efficient preconditioner for the CG method [2,41,42], see Appendix A.3 for a brief insight. Using the PCG method leads to a potentially better convergence rate and speeds up the CG method dramatically, but requires fine-tuning of another parameter.

Let us mention that there exists another class of efficient solvers to approximate the solution of a sparse system of linear equations, called *multigrid methods*, see, e.g., [43]. These methods are predestined to solve elliptic PDEs, however, they also may work fast for other types of PDEs. Multigrid methods and extensions of it can be used as solver itself or also as preconditioner for PCG, cf. [42,44,45]. Due to the large variety of different multigrid methods it is beyond the scope of this work to investigate also the use of those methods here.

4.3. Model Order Reduction (MOR)

The introduced implicit methods have to handle large sparse systems, whereby the computational costs are directly linked to the number of grid points that result from the spatial discretisation. A potential alternative to implicit solvers are MOR techniques, which are frequently used in different research fields such as control system theory, circuit design, fluid mechanics, thermal flow problems or computational biology for reducing the computational complexity of numerical simulations. The basic idea of MOR techniques is to approximate the original high-dimensional ODE system (28) by a very low r-dimensional dynamical system

$$\begin{aligned} \dot{\boldsymbol{u}}_r(t) &= L_r \boldsymbol{u}_r(t) + K_r \boldsymbol{w}(t) \\ \boldsymbol{y}_r(t) &= V \boldsymbol{u}_r(t), \quad \boldsymbol{u}_r(0) = \boldsymbol{u}^{r,0} \end{aligned} \tag{37}$$

with reduced vector $\boldsymbol{u}_r(t) \in \mathbb{R}^r$, projection matrix $V \in \mathbb{R}^{N \times r}$ and $r \ll N$, $L_r \in \mathbb{R}^{r \times r}$, $K_r \in \mathbb{R}^{r \times p}$, $\boldsymbol{y}_r(t) \in \mathbb{R}^N$, $\boldsymbol{u}^{r,0} \in \mathbb{R}^r$ so that the main characteristics of the original ODE system are preserved. The latter procedure is described in more detail in Appendix A.4. In this work, we are generally interested in the complete temperature distribution so that the introduced output vector $\boldsymbol{y}(t)$ is of full dimension (which is a difference to some other works found in the literature).

In fact, MOR methods can be seen as a kind of preprocessing when using time integration methods that reduce computational complexity by reducing the number of describing equations of a large dynamical system. Usually, MOR methods are based on projections and existing techniques can be classified into *balancing-based methods* and *moment-matching methods* or in *SVD-based methods* and *Krylov-based methods*. The classification is not always consistent in the literature, for a general overview see, e.g., [5,46,47].

Let us emphasise that projection-based MOR methods are characterised by the way of constructing the projection matrix. Common and widely used approaches in context of simulation of parabolic PDEs are *modal coordinate reduction* (MCR) [35], *balanced truncation* (BT) [8,10], *proper orthogonal decomposition* (POD) [10,48,49] and KSMOR [7–10].

When dealing with linear large-scale dynamical systems, the KSMOR technique is generally the method of choice, since its construction is based on computationally efficient Krylov subspace methods and is therefore the most efficient MOR method. In the following, a brief introduction is provided, for more details, we refer the reader to Appendix A.5 or, e.g., to [50–53].

4.3.1. Krylov Subspace Model Order Reduction (KSMOR)

The idea of KSMOR methods is to construct a reduced order model such that the moments of the reduced transfer function $\widetilde{\boldsymbol{H}}$ matches the moments of the original transfer function $\boldsymbol{H}$ up to a certain degree. In this context, the transfer function describes the input–output behaviour of the system in the frequency domain.

First of all, let us mention that many MOR methods, also moment-matching techniques, assume zero initial condition, i.e., $\boldsymbol{u}^0 = \mathbf{0}$; therefore, by use of the coordinate transformation $\widetilde{\boldsymbol{u}}(t) = \boldsymbol{u}(t) - \boldsymbol{u}^0$ the original *multi-input-complete-output* (MICO) system (A8) must be translated into

$$\begin{cases} \dot{\widetilde{\boldsymbol{u}}}(t) = L\widetilde{\boldsymbol{u}}(t) + L\boldsymbol{u}^0 + K\boldsymbol{w}(t) \\ \widetilde{\boldsymbol{y}}(t) = \widetilde{\boldsymbol{u}}(t) + \boldsymbol{u}^0, \;\; \widetilde{\boldsymbol{u}}(0) = \mathbf{0} \end{cases} \tag{38}$$

where the non-zero initial condition now appears on the right-hand side of the ODE system, as firstly introduced in [54]. The transformed system (38) with zero initial conditions can be rewritten as

$$\begin{cases} \dot{\widetilde{\boldsymbol{u}}}(t) = L\widetilde{\boldsymbol{u}}(t) + \widetilde{K}\widetilde{\boldsymbol{w}}(t) \\ \widetilde{\boldsymbol{y}}(t) = \widetilde{\boldsymbol{u}}(t) + \boldsymbol{u}^0, \;\; \widetilde{\boldsymbol{u}}(0) = \mathbf{0} \end{cases} \tag{39}$$

where $\widetilde{K} = [K, L\boldsymbol{u}^0]$ and $\widetilde{\boldsymbol{w}}(t) = [\boldsymbol{w}(t), 1]^\top$.

Using the Laplace transform, the transfer function of the original system can be expressed in the frequency domain via $\boldsymbol{H}(s) = (sI - L)^{-1}\widetilde{K}$. As already mentioned, the key idea of KSMOR is based on moment matching. In particular, the moment expansion of the transfer function around a (complex) frequency σ is defined as

$$\boldsymbol{H}(s) = -\sum_{k=0}^{\infty} m_k(\sigma)(s-\sigma)^k \tag{40}$$

where the coefficients $m_k(\sigma) = (L - \sigma I)^{-(k+1)}\widetilde{K}$ are called the *moments* of the transfer function. On this basis, the concept of MOR by means of *moment matching* is to approximate the transfer function (40), in which the intended reduction focuses on matching the first moments around σ. More precisely, the aim is to construct a reduced system (37) so that its transfer function $\widetilde{\boldsymbol{H}}(s) = -\sum_{k=0}^{\infty} \widetilde{m}_k(\sigma)(s-\sigma)^k$ matches the first q moments of the original transfer function:

$$m_k(\sigma) = \widetilde{m}_k(\sigma), \quad k = 0, \ldots, q-1 \tag{41}$$

In this work, we apply the one-sided Arnoldi approach, i.e., $W = V$, which amounts to constructing an orthogonal basis using the input block Krylov subspace

$$\text{range}(V) = \mathcal{K}_q\Big((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\widetilde{K}\Big) \tag{42}$$

Due to the fact that $\widetilde{K} \in \mathbb{R}^{N\times(p+1)}$ is a matrix, the projection matrix $V \in \mathbb{R}^{N\times r}$ with dimension $r = (p+1)q$ is computed by the block Arnoldi algorithm [55] and the reduced system leads to

$$\begin{cases} \dot{\widetilde{\boldsymbol{u}}}_r(t) = L_r\widetilde{\boldsymbol{u}}_r(t) + \widetilde{K}_r\widetilde{\boldsymbol{w}}(t) \\ \widetilde{\boldsymbol{y}}_r(t) = V\widetilde{\boldsymbol{u}}_r(t) + \boldsymbol{u}^0, \;\; \widetilde{\boldsymbol{u}}_r(0) = \mathbf{0} \end{cases} \tag{43}$$

with $L_r = V^\top LV \in \mathbb{R}^{r\times r}$ and $\widetilde{K}_r = V^\top\widetilde{K} \in \mathbb{R}^{r\times(p+1)}$. In summary, the reduced system (43) approximates the original system (A8) in the sense that their first q moments match. Subsequently, the reduced system can be efficiently solved using an implicit time integration scheme (in this work the CN method) via a direct solver and LU-decomposition. After computing the solution of the reduced model, the original solution can be recovered via $\widetilde{\boldsymbol{u}}(t) \approx V\widetilde{\boldsymbol{u}}_r(t)$ and $\boldsymbol{u}(t) = \widetilde{\boldsymbol{u}}(t) + \boldsymbol{u}^0$.

A parameter still to be determined is the choice of the expansion point σ. The value of σ corresponds to the frequencies contained in the original model, so that small values

approximate low frequencies and large σ higher frequencies. The underlying heat evolution is characterised by a rather slow dynamic, therefore approximating the system at the frequency $\sigma = 0$ is a natural choice. In particular, the inverse $(L - \sigma I)^{-1}$ does not exist for $\sigma = \lambda_i$, where λ_i corresponds to an eigenvalue of L.

Treatment of a Large Number of Inputs

A significant weakness of KSMOR arises when dealing with dynamical systems that contain high-dimensional input vectors. To retain high computational efficiency the KSMOR techniques typically require a relatively small number of inputs (small input matrix). The efficiency limitations caused by the large-scale input can be explained as follows: first, a large number of inputs leads to higher computational costs when constructing the Krylov subspace (42). Second, every input eventually needs to be expanded in terms of several moments, so that the order r of the reduced system grows significantly with the number of inputs.

In general, it is assumed that the number of inputs is quite small and usually independent of the fineness of the grid; however, the GES application considered in this work cannot guarantee this feature. In particular, the input size depends on the user-defined dimensions of the heat tank and the incorporated insulating walls, which have to fulfil the matching conditions described in Section 2.2. Based on these considerations, the KSMOR method should not be applied directly and a method adaptation with respect to large-scale inputs is absolutely necessary.

At present, there exists no universal procedure to tackle the problem of a large input dimension, and how to deal with this issue also generally depends on the model problem. Some investigations [9,56–61] have been performed in the past. The proposed methods [56–58,60], see also [62] for an overview, are based on the approximation of the input matrix using the dominant singular vectors of the transfer function at the steady state. Usually, these approaches are limited to systems with a high correlation between the various inputs and outputs, which is not the case for thermal systems as considered here. Another method is based on the superposition principle to linear systems introduced in [59]. The reduction is performed separately using each column of the input matrix so that the original system is approximated by the summation of the output responses of each of the single-input reduced systems. As a result, the original multi-input system is decoupled into separate single-input systems. The superposition approach leads to the fact that the same number of moments is matched and a reduced system of the same order is built. Consequently, in order to attain the same accuracy this technique reduces the computational complexity for systems with a large number of inputs compared to the standard KSMOR method. Nonetheless, the superposition principle for systems with large-scale inputs remains computationally intensive for practical purposes.

The recently proposed approach by [9] is similar to ours. The basic idea is to use an input matrix reduction based on a snapshot matrix linked with computing the dominant singular vectors. We also make use of snapshots here, but in order to reduce the computational costs we neglect the SVD and use the snapshot matrix directly. The idea of using the snapshot matrix directly is applied to a nonlinear input problem in [61], where the authors proposed to use the weighted average of these snapshots to form a single input; however, this is not an appropriate approach for this application.

4.3.2. Modified Krylov Subspace Model Order Reduction (KSMOR*)

Let us now propose a heuristic procedure to resolve the problem of a large number of inputs that incorporates some technical novelties. In doing so, we only adapt the input matrix explicitly for the subspace construction procedure (42). For the description we are returning to the ODE system (28), in particular to the part of the control term. For simplicity,

we represent the p-dimensional input vector $\boldsymbol{w}(t)$ and the corresponding input matrix $K \in \mathbb{R}^{N\times p}$ by

$$\boldsymbol{w}(t) = \big(w_1(t), \ldots, w_p(t)\big)^\top, \quad K = \begin{pmatrix} | & | & & | \\ K_1 & K_2 & \cdots & K_p \\ | & | & & | \end{pmatrix} \tag{44}$$

with $K_i \in \mathbb{R}^N$ for $i = 1, \ldots, p$. As is known, the computational costs for the construction of $V = \mathcal{K}_q\big((L-\sigma I)^{-1}, (L-\sigma I)^{-1}K\big)$ for large values p are immense.

To motivate our proposed heuristic procedure, we assume that the input vector is time-independent, i.e., $\boldsymbol{w}(t) \equiv \boldsymbol{w}$, then the input matrix K from (44) could be assembled via

$$K\boldsymbol{w} =: \widehat{\boldsymbol{K}} \in \mathbb{R}^N \tag{45}$$

which would be accompanied by very low computational costs for generating

$$\widehat{V} = \mathcal{K}_q\Big((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\widehat{\boldsymbol{K}}\Big) \tag{46}$$

However, when executing the latter approach the discrepancy between the subspaces V and $\widehat{V}$ may be large due to the dimensional difference between the matrix K and the vector $\widehat{\boldsymbol{K}}$. In other words, the dimensional difference can be comprehended as a kind of information loss and affects the moment matching process.

Therefore, we modify this idea for simplification by building a compromise between V and $\widehat{V}$, and construct a new input matrix $\overline{K} \in \mathbb{R}^{N\times s}$, where the columns lie in a s-dimensional subspace of K with $s \ll p$. Here, similar to the POD technique, the realisation of this procedure is based solely on *snapshots* from a numerical simulation of the dynamical system. More precisely, using snapshot data

$$W = \big[\boldsymbol{w}(t_1), \boldsymbol{w}(t_2), \ldots, \boldsymbol{w}(t_s)\big] = \begin{pmatrix} w_1(t_1) & w_1(t_2) & \cdots & w_1(t_s) \\ w_2(t_1) & w_2(t_2) & \cdots & w_2(t_s) \\ w_3(t_1) & w_3(t_2) & \cdots & w_3(t_s) \\ \vdots & \vdots & & \vdots \\ w_p(t_1) & w_p(t_2) & \cdots & w_p(t_s) \end{pmatrix} \in \mathbb{R}^{p\times s} \tag{47}$$

generated by a simulated model, we can construct the snapshot-based input matrix $\overline{K}$ and the subspace $\overline{V}$ via $\overline{K} = KW$ and $\overline{V} = \mathcal{K}_q\big((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\overline{K}\big)$, respectively. The original KSMOR method is then continued with K and $\boldsymbol{w}(t)$. This means, the methods KSMOR and KSMOR* differ only in the computation of the subspaces V and $\overline{V}$, which are based on the corresponding input matrices.

It should be noted that for the proposed approach a numerical simulation is necessary in this work, because the temperature vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ within (29) are not known *a priori*. The viability of the modified KSMOR method is demonstrated using a test example in the Appendix A.6.

4.4. Summary of Numerical Solvers

The particular challenge is to find an efficient numerical method in connection with the long-term simulation of a GES. Although many numerical methods have been developed, there is no class of methods that can be considered superior for stiff linear ODE systems, since the correct choice of the method used strongly depends on the underlying model problem. We briefly summarise the main properties of the introduced numerical solvers in Table 2 below, a more detailed summary can be found in Appendix A.7.

Table 2. Summary of numerical solvers presented in Section 4, focusing on classic explicit (EE) and implicit (CN) methods as well as two popular techniques, namely the fast explicit methods (FED and FSI) and the Krylov-based model order reduction techniques (KSMOR and KSMOR*).

Solver	Pros	Cons
EE	• Explicit method • Easy to implement and well-suited for parallel computing	• Suffers from severe time step size restriction to satisfy numerical stability, therefore impractical for stiff ODEs
CN	• Implicit method • No time step size restriction • Linear systems can be efficiently solved by a factorisation/preconditioning technique (precomputed once only)	• Requires the solution of many large sparse linear systems (updating external sources at relatively small time intervals), therefore results in high computational effort
FED	• Accelerated explicit method • Easy to implement and well-suited for parallel computing	• Highly sensitive to numerical rounding errors • Not practical for extremely stiff problems
FSI	• Accelerated explicit method • Easy to implement and well-suited for parallel computing • Ensures (numerical) internal stability	• Not practical for extremely stiff problems
KSMOR	• Stability preservation (one-sided projection method) • Typically, only a small dimension is required to construct an accurate reduced order model in a highly efficient manner	• Not practical for a large number of inputs
KSMOR*	• Stability preservation (one-sided projection method) • Typically, only a medium dimension is required to construct an accurate reduced order model in a highly efficient manner • Applicable for models with many inputs	• Globally accurate solutions are provided only for a reasonable high number of snapshots and Krylov subspaces

5. Comparison of Solvers for Long-Term GES Simulation

The numerical long-term simulation of the GES model considered in this work, is mainly focused on the explicit FSI method, the implicit sparse direct/iterative solver and the KSMOR* technique, introduced in the section before. In the following, we evaluate the proposed numerical solvers by two different experiments and assess their performance in terms of approximation quality and *central processing unit* (CPU) time. As methods of choice we consider:

1. EE method. An estimation of the upper bound $\tau_{\max}$ is necessary.
2. FSI scheme. By knowing $\tau_{\max}$, the number M of cycles is the only tuning parameter.
3. Sparse direct solver applied to the implicit CN method (called direct CN). The internal MATLAB-function *decomposition* including a Cholesky factorisation is used. The tuning parameter is the time step size τ.
4. Sparse iterative solver using PCG applied to the implicit CN method (called iterative CN). User-defined parameters are time step size τ and the tolerance $\varepsilon > 0$; the latter quantity is required to terminate the algorithm. The preconditioners IC or MIC are employed for PCG. Another tuning parameter $\gamma > 0$ is required, which corresponds to a numerical fill-in strategy IC(γ) or MIC(γ).
5. KSMOR* method. This technique can be tuned by the number of projection subspaces $\mathcal{K}_q$ and snapshot data $W = \left[\boldsymbol{w}(t_1), \boldsymbol{w}(t_2), \ldots, \boldsymbol{w}(t_s)\right]$ used. The selected expansion point is fixed to $\sigma = 0$. The sparse direct solver is applied to compute the Krylov subspace generated by the block Arnoldi method. The reduced model is then solved via direct CN scheme for a chosen τ.

To evaluate the accuracy of the methods, the solution of the EE scheme is used as reference. Based on a simplified GES application without a source term, we first examine the optimal selection of some important parameters within the methods used. Second, we validate the method performances on real data including sources. In addition, a uniform grid with $\Delta x = \Delta y = h$ is considered in all experiments and the evaluation is measured by the L_2-error defined as

$$L_2(\boldsymbol{u}, \widetilde{\boldsymbol{u}}) = \sqrt{\sum_{i=1}^{N} (u_i - \widetilde{u}_i)^2} \tag{48}$$

between the reference solution $\boldsymbol{u}$ and the numerical solution $\widetilde{\boldsymbol{u}}$.

Let us mention that to evaluate the accuracy of the methods, we also consider a simple artificial setup in Appendix B, where the solution can be stated in closed form. Furthermore, we note that all experiments were performed in MATLAB R2018b with an Intel Xeon(R) CPU E5-2609 v3 CPU. All CPU times presented incorporate the modelling (linear system, preconditioning and reduction) and the numerical resolution; therefore, the performances are easily comparable.

5.1. Geothermal Energy Storage Simulation without Source

First, we consider the simulation of a GES under real-world conditions without sources/sinks. The heat tank is assumed to be placed underground, closed on its sides and upwards, compare Figure 2. The lower part is open to accommodate for the heat delivery. The thermophysical properties of the materials used are assumed to be constant and are given in Table 3. In this setup, we specify that the upper and the lower ground have the same material properties, and further parameters are fixed to $\alpha_c = 0.5$ and $\alpha_A = 10$. As mentioned in Section 2.3, the external conditions are given via time-dependent Robin (6), time- and space-dependent Dirichlet (7) and non-homogeneous Dirichlet boundary conditions (9) with $\widetilde{g} = 10$.

Table 3. Thermophysical properties of the materials involved for the synthetic experiment.

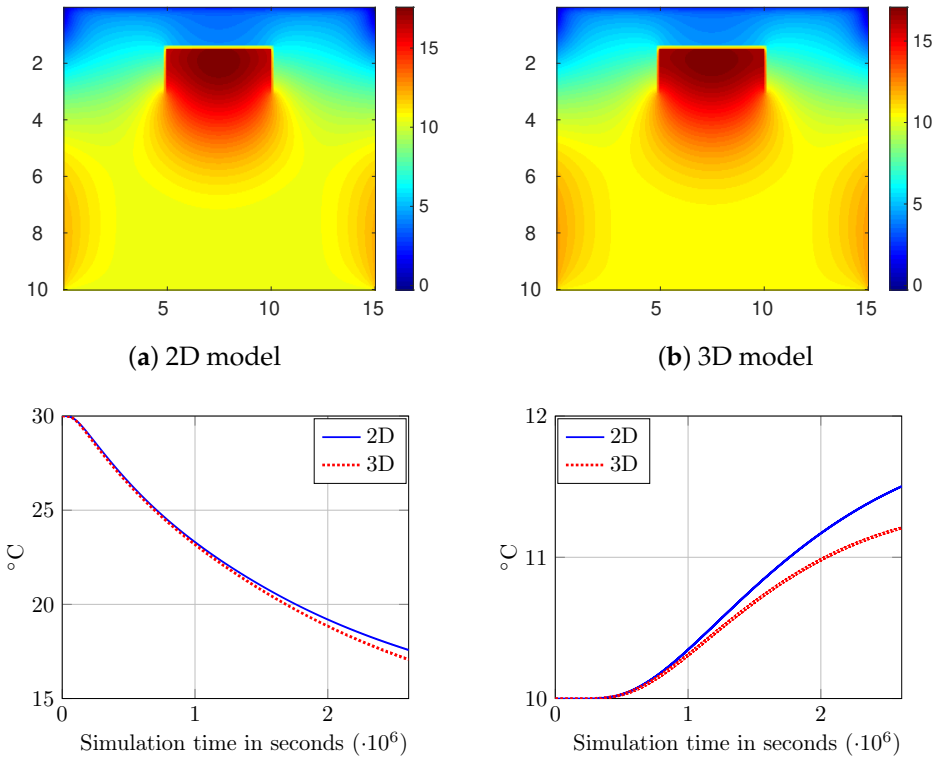
Material/Layer	Conductivity λ (W/(mK))	Density ρ (kg/m^3)	Specific Heat c (J/(kgK))	Diffusivity a (m^2/s)
Water-saturated soil	2.3	2100	1143	9.58×10^{-7}
Insulation walls	0.03	100	1000	3×10^{-7}

For the evaluation, a two-dimensional rectangular domain is considered given by $[0, 15] \times [0, 10]$ in meters, whereby the heat tank with size 5×1.5 m has an installation depth of 1.5 m. The thickness of the insulation is determined to 12 cm. Moreover, a mid-size model problem is considered by setting $h = 0.04$ cm ($N = 94376$ grid points) and the stopping time is assumed to be $t_F = 2.609 \times 10^6$ s (around 30 days). As initialisation, the temperature is simply fixed to 30 degrees for the heat tank, 20 degrees for the insulation and 10 degrees otherwise.

Let us first discuss the spatial dimensioning of the underlying model. The original geothermal field is given in 3D, but it is clear that its fundamental simulation is computationally more expensive than a two-dimensional model. In fact, however, a full three-dimensional simulation is not necessary, which can be explained as follows: the considered model problem is isotropic, homogeneous, linear and characterised by an almost symmetrical structure (concerning the spatial environment) with slow heat propagation (low diffusivity). This allows the physical behaviour of the three-dimensional continuous model to be approximated by considering a cross-section of a 3D geothermal field in 2D, especially when dealing with long-term evolutions. The proposed heuristic approach can be theoretically supported by a closer look at the fundamental solution (heat kernel) of the linear heat equation. More precisely, the fundamental solution is radial in the spatial variable, but depends on a dimensional-dependent scaling factor, see, e.g., [63]. Aside from the purely physical justification, we confirm the reduction in the spatial model dimensions by comparing the results in 2D and 3D using the EE method. Further in the paper, we show that the approach is justified by taking real data into account.

Results on EE

According to Gershgorin's circle theorem the time step size restriction for the two-dimensional model problem is given by $\tau_{\max,2D} \approx 417.44$ s. Consequently, 6250 iterations have to be performed to reach the final stopping time t_F, whereby the corresponding CPU time of EE amount to 11.1 s. In contrast, the upper stability bound of the underlying three-dimensional problem is determined with $\tau_{\max,3D} \approx 278.15$ so that 9380 iterations for the simulation are necessary. In this case, the discrete domain is defined by $N = 35,485,376$ grid points, which enlarges the size of the system matrix considerably and thus increases the computational effort. Consequently, the entire simulation requires around 20,000 s which is considered unsatisfactory. As expected, Figure 4 confirms the use of a two-dimensional model as a substitute. First, it shows almost equal radial thermal behaviour when comparing the 2D and 3D models, in which the latter is evaluated via a cross-section along the middle of the 3D-GES. Second, a comparison of two artificial temporal temperature profiles S1 and S2 demonstrate the expected similarities of the 2D and 3D simulation results. Of course, there are slight deviations in heat evolution between the 2D and 3D models, but the benefit of computational performance is more valuable than computing an extremely accurate 3D long-term simulation. Overall, we consider the dimensionality reduction in the model as a pragmatic proceeding to obtain fast and accurate enough long-term simulations of typical GES. In the further course, we compare the proposed methods using the 2D-GES model on both synthetic and real data.



(**a**) 2D model (**b**) 3D model

(**c**) Temporal temperature of S1 (**d**) Temporal temperature of S2

Figure 4. Results of the GES experiment without source: visual representation of the 2D and 3D solution at $t_F = 2.609 \times 10^6$ by applying the EE method for $h = 0.04$. (**a**) Result of the 2D model after 6250 iterations with $\tau_{\text{max,2D}} = 417.44$. (**b**) Result of the 3D model after 9380 iterations with $\tau_{\text{max,3D}} = 278.145$, visualised in form of a cross-section along the middle of the 3D-GES. (**c**) Temporal temperature profile S1 fixed in the centre of the heat tank. (**d**) Temporal temperature profile S2 fixed in the centre between heat tank and bottom border.

Let us emphasise that for the following evaluations of the solvers no analytical solution is available and a reference solution is required. In general, the EE method provides the most precise solution (cf. Appendix B) that can be used as reference solution for a fair comparison. To this end, we assess the L_2-error at a fixed moment in time and the corresponding CPU time between the solutions of EE and the tested solver for the experiment without source term. In doing so, we do not consider a full error computation along the temporal axis because the methods yield completely different sampling rates, which potentially leading to unfair comparisons.

Results on Iterative CN

We now study the iterative solver and its parameters involved. For this purpose, the influence of the stopping criterion on the accuracy and the CPU time is analysed in Figure 5. The value $\varepsilon = 10^{-4}$ provides the best trade-off and is assumed to be fixed for all following tests. Due to the size of the system matrix a preconditioning is suitable. By consideration of the specified preconditioners, Figure 6 shows that MIC(10^{-2}) outperforms both CG and IC(10^{-3}). Based on this investigation, we use MIC(10^{-2}) for all further experiments.

To demonstrate the beneficial applicability of the CN method in general, we compare its results with those of IE in Appendix C.1.

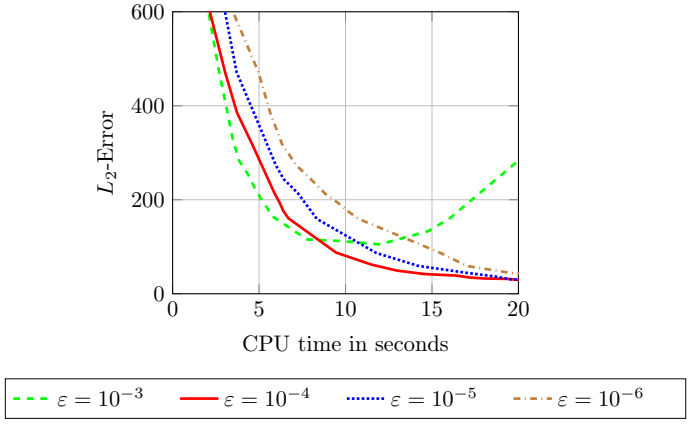


Figure 5. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time for iterative CN using the CG method and different values ε at $t_F = 2.609 \times 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. The choice $\varepsilon = 10^{-4}$ provides the best trade-off.

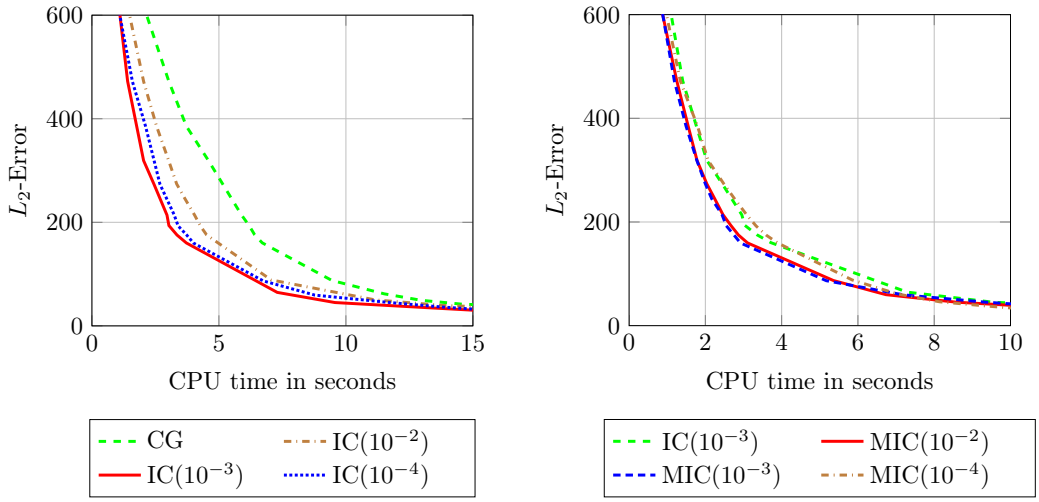


Figure 6. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time using iterative CN at $t_F = 2.609 \times 10^6$ for $h = 0.04$. The stopping criterion is fixed to $\varepsilon = 10^{-4}$. The reference solution is computed with the EE method. **Left:** Iterative CN is computed using IC-preconditioner for different drop tolerances γ. The choice $\gamma = 10^{-3}$ provides the best trade-off. **Right:** Iterative CN is computed by IC(10^{-3}) and MIC-preconditioner for different drop tolerances γ. A value for $\gamma \in [10^{-3}, 10^{-2}]$ is optimal.

Results on KSMOR*

Due to external, lateral time- and space-dependent Dirichlet boundary conditions and internal matching conditions, the model has to deal with many inputs. In particular, the input variable results in $\widetilde{\boldsymbol{w}}(t) \in \mathbb{R}^{1106}$, with $\boldsymbol{u}_2(t) \in \mathbb{R}^{852}$ and $\boldsymbol{T}_g(t) \in \mathbb{R}^{251}$, while $\boldsymbol{u}_1$ does not have to be taken into account due to the identical material properties of both soils (upper and lower ground).

To overcome the efficiency limitation of KSMOR (see Appendix C.2) we apply the KSMOR* approach, proposed in Section 4.3.2. In doing this, we initially set $q = 2$ and vary the size of the snapshot data as visualised on the left in Figure 7. Obviously, increasing s leads to better approximations; however, a saturation occurs for high values s, for which $s = 20$ provides the best performance trade-off. In a second test, we increase the number of subspaces q for $s = 20$ used, see on the right in Figure 7. As expected, increasing the moment matching property improves also the accuracy. Unfortunately, large values q

are associated with high computational costs. Overall, KSMOR* is much more efficient compared to the original KSMOR method (cf. Appendix C.2).

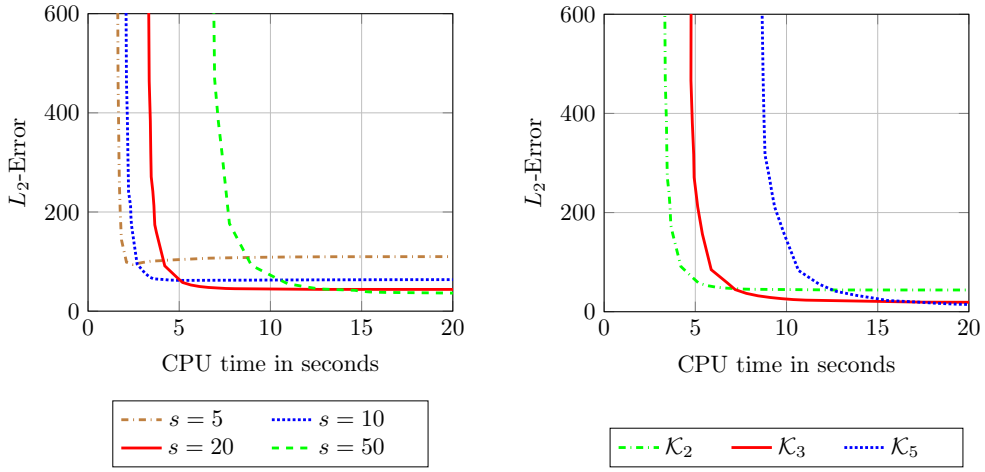


Figure 7. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time using KSMOR* at $t_F = 2.609 \times 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. **Left:** KSMOR* is applied with $q = 2$ and a varying size of snapshot data for $s = \{5, 10, 20, 50\}$. **Right:** KSMOR* is used by constructing subspaces with $s = 20$ for $q = \{2, 3, 5\}$. The best compromise in performance is achieved with $s = 20$ and $q = 3$. Compared to the original KSMOR method (cf. Appendix C.2), the approximations are computed dramatically fast.

5.1.1. Comparison of the Solvers

Finally, a full comparison of the methods tested—direct CN, iterative CN using MIC(10^{-2}), FSI and KSMOR*—is shown in Figure 8.

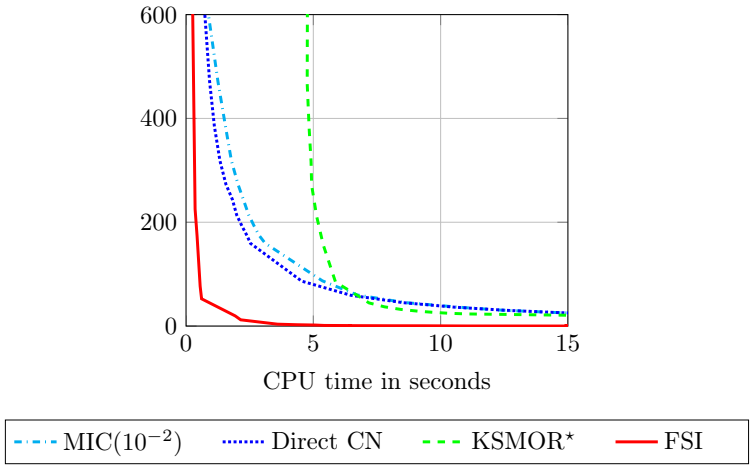


Figure 8. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time between MIC(10^{-2}), direct CN, KSMOR* and FSI at $t_F = 2.609 \times 10^6$ for $h = 0.04$. KSMOR* is performed with $q = 3$ and $s = 20$. The reference solution is computed with the EE method. The FSI method clearly outperforms all other methods.

Solving the CN method with the sparse direct solver is more efficient than with the iterative solver. Only for more accurate solutions, which correspond to small time step sizes τ, both implicit solvers perform comparably well due to a faster convergence of the iterative solver.

The KSMOR* technique, in which the parameters $s = 20$ and $q = 3$ are fixed, achieves the worst performance in comparison to the other methods. In contrast to the investigated test example in Section 4.3.2, the selected number of snapshots has to be relatively large. In combination with the number of Krylov subspaces directly related to expensive

offline precomputations and the additional enlargement of the reduced solution (A15), the KSMOR* approach cannot maintain the efficiency of the other solvers.

The results in Figure 8 clearly demonstrate the superior efficiency of FSI, which outperforms all other methods. This is mainly explained by two facts: first, FSI is an accelerated explicit method and is built on cheap matrix-vector multiplications. Second, the input $\boldsymbol{w}(t)$ can be updated within one FSI-cycle. Further, it is this update process, including the external and internal boundary conditions, that is of great importance for an accurate approximation. In this context, we also show a performance comparison of the two fast explicit solvers FED and FSI in Appendix C.3.

We would like to point out once again that the generated results are based on a mid-size model problem. In Appendix C.4, we additionally present that the same solver performances are preserved for a large-scale problem for which we repeat this experiment on a finer grid.

5.1.2. Observations on KSMOR*

From an engineering point of view a more detailed investigation of the approximation behaviour regarding the underlying model shows that small values s and q are deemed appropriate. A visualisation of the results between the absolute differences of EE and KSMOR* for $h = 0.04$ and a different number of s and q is shown in Figure 9. As expected, increasing the number of snapshots s or subspaces q leads to better results in terms of the maximum error and the L_2-error. An individual change of s and q leads to a kind of saturation behaviour, so that both parameters require relatively large values for a very accurate approximation.

Moreover, a reasonable number of snapshots s is significantly important because less data lead to strong artefacts at the interfaces where the matching conditions between different materials have to be fulfilled. This observation is a consequence of the proposed input matrix reduction, which clarifies the interdependence between snapshot data and the errors at the interfaces. Due to the fact that small values of s and q only influence the area around the interfaces (cf. Figure 9), which is generally negligible for a reasonable reproduction of the long-term GES simulation, the use of KSMOR* appears to be significantly appropriate.

Based on the presented findings, we investigate the performances of FSI, direct CN and KSMOR* with $s = 10$ and $q = 2$ on real data.

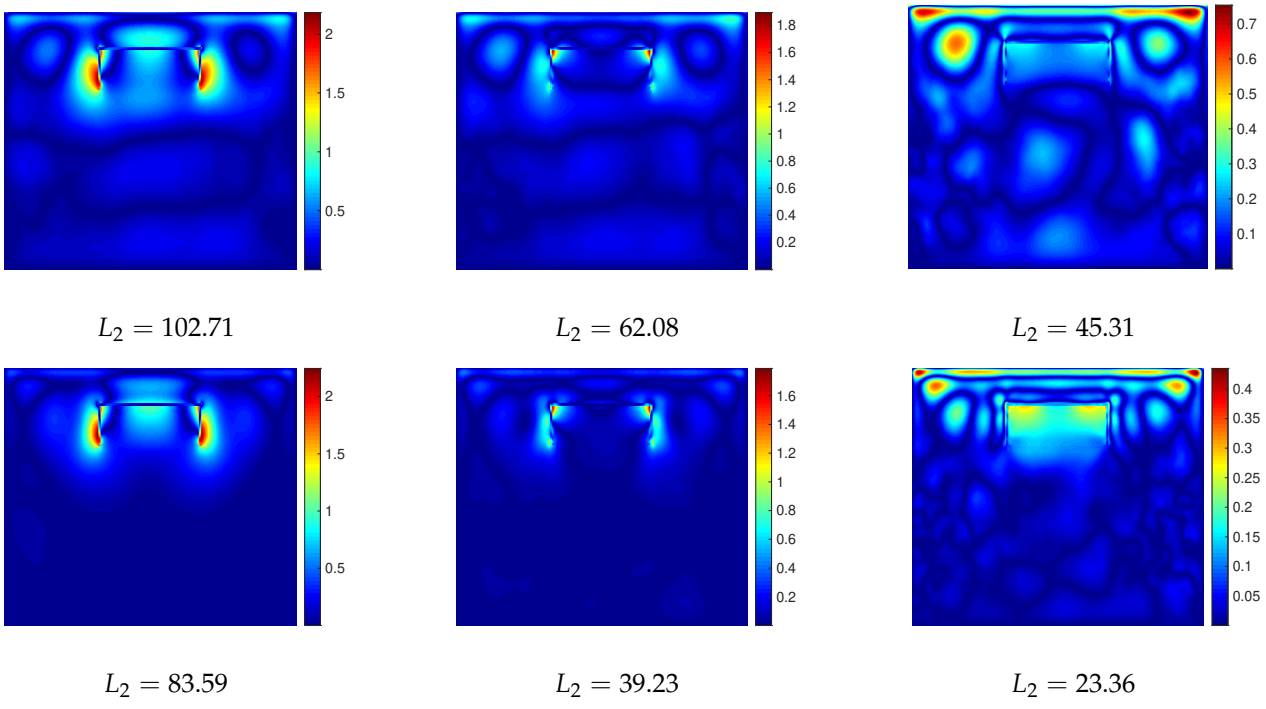


Figure 9. *Cont.*

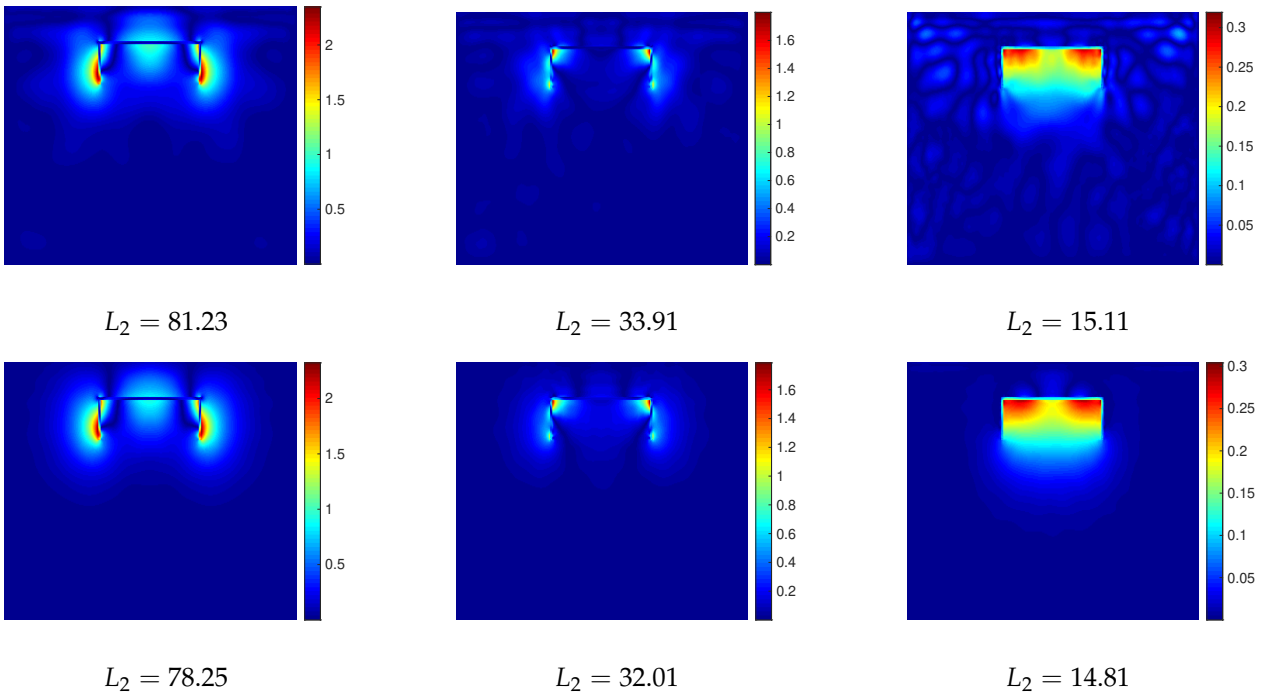


Figure 9. Results of the two-dimensional GES experiment without source: visual comparison (absolute differences) of the results between EE and KSMOR* for varying number of s and q at $t_F = 2.609 \times 10^6$ for $h = 0.04$ with $\tau = 2609$. The visualisations are ordered **from left to right** by $s = \{5, 10, 20\}$ and **from top to bottom** via $q = \{2, 3, 5, 10\}$. Special care must be taken in rating the visualisations, because no uniform temperature scaling between the absolute differences of the solutions is given: we still opted to represent the solutions in this way, as the differences between the errors that occur are too large to employ a uniform scaling. For a better comparison, the L_2-error is also displayed. Increasing the number of snapshots s or subspaces q gives better results in terms of maximum error and L_2-error. Obviously, a reasonable selection of s is of great importance, since less snapshot data lead to strong artefacts at the interfaces.

5.2. Geothermal Energy Storage Simulation on Real Data

Lastly, we present a comparison concerning a full error computation along the temporal axis using the two-dimensional GES simulation including sources. The evaluation is based on real data, in particular matching of temperature probes of a test field and its thermal behaviour given in 3D, cf. Figure 10. In this way, the experiment will also demonstrate that the proposed linear two-dimensional model is sufficient to reproduce the heat exchange correctly.

For this experiment, the thermophysical properties of the materials used are assumed to be constant and are given in Table 4. The other parameters are fixed to $\alpha_c = 0.1$, $\alpha_A = 10$ and $\widetilde{g} = 10$. The two-dimensional rectangular domain is defined via $[0, 20] \times [0, 8]$ in meters, whereby the size of the heat tank amounts to 10×1.2 m. The installation depth and the insulation thickness are determined as 70 and 12 cm, respectively. In addition, the thermal energy

$$Q = mc\Delta T \tag{49}$$

with the specific heat capacity c, the mass m of the fluid and temperature change of the inlet and return temperatures ΔT is given over a period of $t_F = 21 \times 10^6$ s (around 243 days). The parameters are fixed to $c = 1.16$ and $m = 850$ and the source term is simply distributed over three pipe levels inside the heat tank. Moreover, the mesh size is fixed to $h = 0.04$ ($N = 100,701$ grid points), which leads to the upper stability bound $\tau_{\text{max,2D}} \approx 588.23$. The initial heat distribution is generated by the Laplace interpolation (see Appendix D.1).

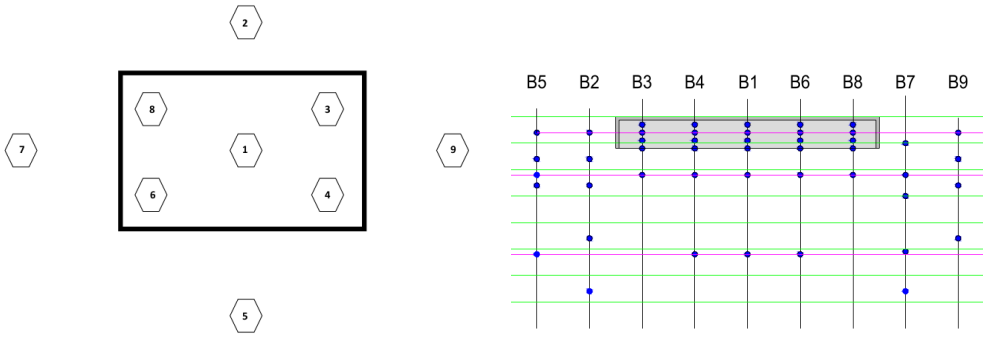


Figure 10. Position of the temperature probes B1-B9 of the 3D test field. **Left:** Plan view of the GES (*rectangular* shape) and the installation location of the temperature probes B1-B9 (*hexagons*). **Right:** Cross-section of the 3D-GES and installation height of the temperature probes B1-B9. Each probe has several measuring points (*blue* dots on *black* vertical lines) at different installation heights. In particular, the measuring points do not always have the same installation height, as indicated by the *magenta* horizontal lines.

Table 4. Thermophysical properties of the materials involved for the real-world data scenario from the present test field.

Layer	Conductivity λ (W/(mK))	Density ρ (kg/m^3)	Specific Heat c (J/(kgK))	Diffusivity a (m^2/s)
Lower ground	0.5	1900	750	3.51×10^{-7}
Upper ground	1.7	2000	1250	6.8×10^{-7}
Insulation walls	0.035	40	1500	5.83×10^{-7}

In order to validate the correct heat exchange behaviour, we initially apply the EE method to the two-dimensional GES model. The visualisation of the real temperature probe B1 (measuring point 3rd from the top, see Figure 10) compared to EE is shown exemplary on the top left in Figure 11. The result clearly demonstrates the appropriate use of a two-dimensional simulation of a 3D-GES model. Note that one may use any working probe to come to the same conclusion.

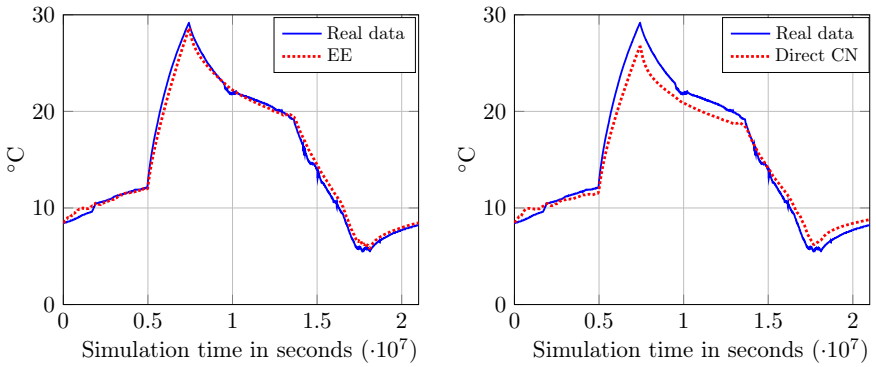


Figure 11. *Cont.*

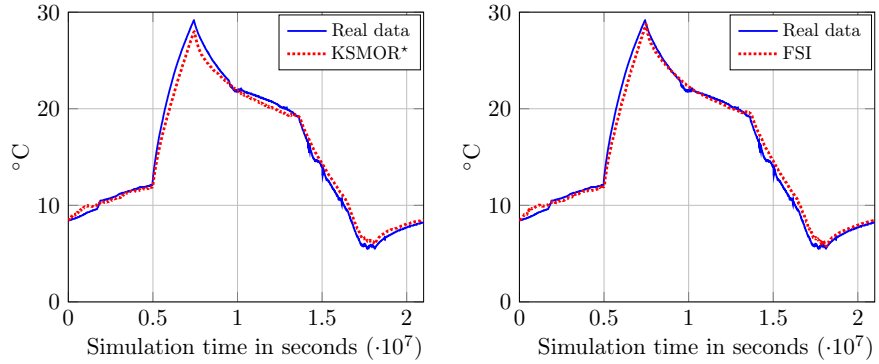


Figure 11. Results of the GES experiment for temperature probe B1 (3rd from the top): visual comparison (along the temporal axis) between the real temperature from 3D test field and the approximations of the 2D model computed by the proposed solvers. A special emphasis on this is that the results generated by FSI, direct CN and KSMOR* cause *CPU time of 10 seconds*, whereas the EE scheme produces substantial *costs of 69 seconds*. The simulation time amounts to around 243 days and the grid size is fixed to $h = 0.04$ ($N = 100,701$ grid points). **Left top:** EE for $\tau_{\text{max,2D}} = 588.23$ (35,839 iterations). **Right top:** Direct CN for $\tau = 44,854$ (470 iterations). **Left bottom:** KSMOR* with $s = 10$ and $q = 2$ for $\tau = 13,176$ (1600 iterations). **Right bottom:** FSI for $M = 220$ cycles (4840 iterations). The EE method applied to the two-dimensional model problem reproduces a highly accurate approximation with regard to real data given from the three-dimensional test field. FSI and KSMOR* are extremely efficient and achieve almost the same output as EE. On closer inspection, the FSI scheme provides the best trade-off as it is less cost-intensive due to matrix-vector multiplications and additionally produces a better accuracy based on a more frequent update process. In contrast, the direct CN method cannot maintain the performance of the other two methods.

Building on the validation, the performances of FSI, direct CN and KSMOR* with $q = 2$ and $s = 10$ are identified using the temperature probe B1 by two variants. First, a visual assessment of the approximations compared to real data is illustrated in Figure 11. In the experiment documented in Figure 11 we give quickly computed approximations (CPU time of 10 s) in comparison with EE (CPU time of 69 s). Second, the relation between the L_2-error (along the temporal axis) and the corresponding CPU time is evaluated in Figure 12. Additional data for comparison of the findings described are listed in Appendix D.2. In both investigations, the FSI scheme is still considered as a superior method, which provides a fast computation combined with high accuracy. Nevertheless, the KSMOR* method is also very efficient. As in the previous tests, the direct CN method provides the worst performance.

Finally, FSI and KSMOR* are tested on the temperature probe B6 (5th from the top) and B9 (2nd from the top), see Figure 13. Both schemes achieve the desired reproduction of heat distribution behaviour compared to the real data; however, the KSMOR* results may also exhibit oscillations, especially in a local area around the interfaces between different materials. This illustrates again that the numerical solutions obtained by KSMOR* are locally representative away from the interfaces if q and s are chosen in a competitive range in our application, whereas the FSI method correctly reproduces the global behaviour of the underlying GES model.

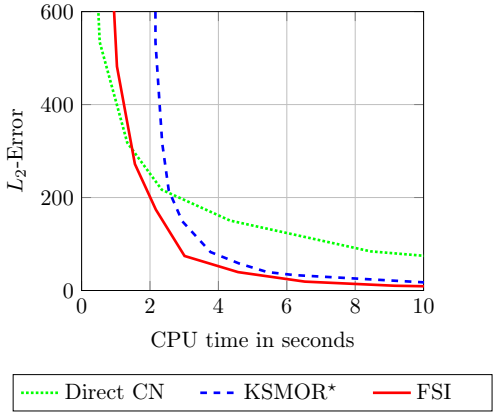


Figure 12. Results of the GES experiment for temperature probe B1 (3rd from the top): comparison of the L_2-error (along the temporal axis) and the corresponding CPU time between FSI, direct CN and KSMOR* with $q = 2$, $s = 10$. The parameters are fixed to $t_F = 21 \times 10^6$ and $h = 0.04$ ($N = 100,701$ grid points). The reference solution is computed with EE. The FSI scheme is the most efficient solver.

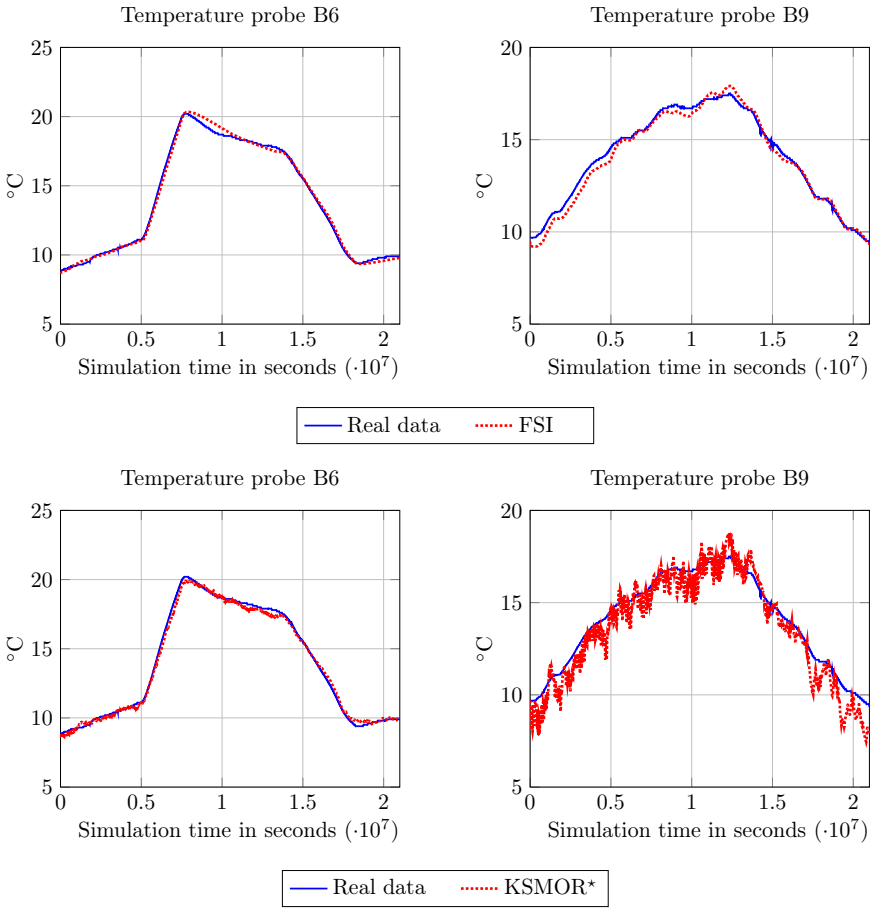


Figure 13. Results of the GES experiment for temperature probe B6 (5th from the top) and temperature probe B9 (2nd from the top) on the **left** and **right** side, respectively: visual comparison (along the temporal axis) between the real temperature from 3D field and the approximations of the 2D model computed by FSI as well as KSMOR*. The CPU time of both computed approximations amount again to 10 seconds. **Top:** FSI for $M = 220$ cycles (4840 iterations). **Bottom:** KSMOR* with $s = 10$, $q = 2$ for $\tau = 13,176$ (1600 iterations). Both methods reproduce the same temperature behaviour compared to the real data; however, the corresponding temperature of the KSMOR* technique can oscillate highly sensitive, especially in a local area around the interfaces between different materials.

6. Summary and Conclusions

In this work, we have demonstrated that a model based on a linear heat equation equipped with external and internal boundary conditions is suitable to realistically represent the long-term behaviour of the GES. Moreover, we showed experimentally based on real-world data from a three-dimensional test field that even a two-dimensional GES simulation is sufficient to tackle the long term simulation task. As a result, the computational costs can be extremely reduced and a long-term simulation is practicable.

In view of the possible candidates for an efficient numerical simulation, a state-of-the-art method that appears to be attractive is the KSMOR scheme. In our model problem we have seen that this scheme is not easy to apply, since the underlying semi-discretised model is linked with a large input vector in consequence of the modelled boundary conditions. Independent of the numerical approach, it has also been noticed that the presence of sources and boundary conditions makes the long-term simulation issue in practice delicate to handle. With these fundamental difficulties, we have illuminated in detail that it is not straightforward to device an efficient and accurate enough numerical scheme.

In total, we have demonstrated the practical usability of the FSI scheme and the KSMOR variant introduced here as KSMOR*, which turn out to be the two most powerful methods among the schemes in this paper for our GES application. The explicit FSI scheme borrowed from image processing, is highly efficient due to cost-effective matrix-vector multiplications and the natural frequent update process of the input vector within the approach. In addition to this, it is by the construction of the FSI method, which can also use modern parallel architectures such as GPUs, that we can further increase the efficiency. Let us also note here that to our best knowledge, our paper provides the first application of FSI outside of the field of image processing. Our proposed efficient KSMOR* technique uses an input-matrix reduction via snapshots and generates a small-sized reduced order model, which can then be resolved easily using the direct solver. We illustrated and analysed the viability of KSMOR*, which is in this form a new variant of existing schemes. At this point it, should be stressed that compared to previous works in the area of KSMOR schemes, we have made our method here explicit in all the details and important parameters, which is by the computational experience gained in the course of this work highly relevant for practical application.

Apart from the fact that FSI and KSMOR* are predestined solvers for tackling the long-term simulation of a GES, we have specifically discussed the local and global behaviour of their solutions. Summarising our findings, we (absolutely) suggest the use of the FSI scheme when the global behaviour of solutions is of interest. Let us also mention that this may be of preference in the context of our application, since this takes into account general configurations, also away from interfaces, to measure temperatures by probes in realistic environments. Otherwise, for local areas but away from interfaces, both techniques perform equally effective.

Overall, we have precisely analysed the features of all applied solvers and simultaneously illustrated their properties using different experiments. For these reasons the comprehensive work given here provides, from our point of view, a reasonable overview of the state-of-the-art numerical solvers of various scientific areas and may help greatly in approaching similar problems in many engineering fields.

Author Contributions: Conceptualization , M.B. (Martin Bähr) and M.B. (Michael Breuß); methodology, M.B. (Martin Bähr) and M.B. (Michael Breuß); software, M.B. (Martin Bähr); validation, M.B. (Martin Bähr); formal analysis, M.B. (Martin Bähr) and M.B. (Michael Breuß); investigation, M.B. (Martin Bähr); resources, M.B. (Martin Bähr) and M.B. (Michael Breuß); data curation, M.B. (Martin Bähr); writing—original draft preparation, M.B. (Martin Bähr) and M.B. (Michael Breuß); writing—review and editing, M.B. (Martin Bähr) and M.B. (Michael Breuß); visualization, M.B. (Martin Bähr); supervision, M.B. (Michael Breuß); project administration, M.B. (Michael Breuß); funding acquisition, M.B. (Michael Breuß). All authors have read and agreed to the published version of the manuscript.

Funding: We would like to thank the funding by the *AiF Projekt GmbH* (ID number: KF3137502NT4).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The code used to generate the results in this work is available from the corresponding author on sincere request.

Acknowledgments: This work was supported by *hartig & ingenieure GmbH*. This includes in particular providing technical information and corresponding real data of a three-dimensional GES. Furthermore, we thank Ralf Wunderlich for helpful discussions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

BT	balanced truncation
CG	conjugate gradient
CN	Crank–Nicolson
CPU	central processing unit
EE	explicit Euler
FED	fast explicit diffusion
FSI	fast semi-iterative
GES	geothermal energy storage
GPU	graphics processing unit
IC	incomplete Cholesky
IE	implicit Euler
KSMOR	Krylov subspace model order reduction
LU	lower-upper
MCR	modal coordinate reduction
MIC	modified incomplete Cholesky
MICO	multi-input-complete-output
MOR	model order reduction
ODE	ordinary differential equation
PCG	preconditioned conjugate gradient
PDE	partial differential equation
POD	proper orthogonal decomposition
SVD	singular value decomposition

Appendix A. Numerical Solvers

Appendix A.1. Fast Explicit Diffusion

The key idea of the technique behind FED is to explore the connection between iterated box filtering and diffusion filtering. Grewenig et al. showed that a box filter of length $2n+1$ can be obtained by convolving n explicit linear diffusion filters having different time step sizes τ_i. These varying time step sizes can be calculated by means

$$\tau_i = \frac{h^2}{2} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)}, \quad i = 0, 1, \ldots, n-1 \tag{A1}$$

and one FED cycle is given as

$$\begin{aligned} \boldsymbol{u}^{k+1,0} &= \boldsymbol{u}^k, \\ \boldsymbol{u}^{k+1,i+1} &= (I + \widetilde{\tau}_i L)\boldsymbol{u}^{k+1,i} + \widetilde{\tau}_i K \boldsymbol{w}^k, \quad i = 0, 1, \ldots, n-1 \\ \boldsymbol{u}^{k+1} &= \boldsymbol{u}^{k+1,n} \end{aligned} \tag{A2}$$

with $\widetilde{\tau}_i := c\tau_i$ and $c := \frac{4}{h^2 \lambda_{max}}$, and where the input vector $\boldsymbol{w}^k$ is constant within the full cycle. At this, the value $\lambda_{\max}$ denotes the largest modulus of eigenvalues. The scaling of τ_i

via c is necessary, since the stability in the Euclidean norm is only guaranteed for $\lambda_{\max} \leq \frac{4}{h^2}$. In FED up to half of all the time steps sizes may violate their upper bound, and n steps can achieve a stopping time of $\mathcal{O}(n^2)$ instead of the usual $\mathcal{O}(n)$ for a valid constant time step size. This gain in the stopping time can result in cutting the overall run time by two orders of magnitude. More precisely, one can show that the diffusion time with n inner steps corresponds to one super time step of size $\sum_{i=0}^{n-1} \tau_i = \tau_{\max} \frac{n^2+n}{3}$ for $\tau_{\max} := \frac{2}{\lambda_{\max}}$. Additionally, increasing the number of cycles, whereby n becomes smaller, improves the accuracy of this method. The only requirement for one FED cycle (A2) is that the system matrix L has to be symmetric, negative semi-definite and remains constant within one cycle for nonlinear problems. Let us stress that FED is often used in conjunction with homogeneous nonlinear and anisotropic diffusion filtering; however, the scheme can be applied to many parabolic problems including sources/sinks, cf. [64].

Appendix A.2. Krylov Subspace Solvers

A particular class of iterative solvers designed for use with large sparse linear systems is the class of *Krylov subspace solvers*; for a detailed exposition see [2]. The main idea behind the Krylov approach is to search for an approximative solution of (35) in a suitable low-dimensional subspace $\mathbb{R}^l$ of $\mathbb{R}^N$ that is constructed iteratively with l being the number of iterates. The aim in the construction is thereby to have a good representation of the solution after a few iterates.

Conjugate Gradient Method

The CG method [39] is probably the most famous Krylov subspace method and a widely used iterative solver for problems with large, sparse symmetric and positive definite matrices. In its construction it combines the gradient descent method with the method of conjugate directions. It is based on the fact that for such a matrix the solution of $A\boldsymbol{x} = \boldsymbol{b}$ is exactly the minimum of the function

$$F(\boldsymbol{x}) = \frac{1}{2}\langle \boldsymbol{x}, A\boldsymbol{x}\rangle_2 - \langle \boldsymbol{b}, \boldsymbol{x}\rangle_2 \tag{A3}$$

since

$$\nabla F(\boldsymbol{x}) = \boldsymbol{0} \quad \Leftrightarrow \quad A\boldsymbol{x} = \boldsymbol{b} \tag{A4}$$

Thereby, $\langle \cdot, \cdot \rangle_2$ means the Euclidean scalar product. Let us now denote the l-th Krylov subspace by $\mathcal{K}_l$. Then, $\mathcal{K}_l := \mathcal{K}_l(A, \boldsymbol{r}_0)$ is a subspace of $\mathbb{R}^N$ defined as

$$\mathcal{K}_l := \mathsf{span}\left(\boldsymbol{r}_0, A\boldsymbol{r}_0, A^2\boldsymbol{r}_0, \ldots, A^{l-1}\boldsymbol{r}_0\right) \tag{A5}$$

This means $\mathcal{K}_l$ is generated from an initial residual vector $\boldsymbol{r}_0 = b - A\boldsymbol{x}_0$ by successive multiplications with the system matrix A. The nature of an iterative Krylov subspace method is that the computed approximate solution $\boldsymbol{x}_l$ belongs to $\boldsymbol{x}_0 + \mathcal{K}_l(A, \boldsymbol{r}_0)$, i.e., it is determined by the l-th Krylov subspace. Thereby, the index l is also the l-th iteration of the iterative scheme.

For the CG method, one can show that the approximate solutions $\boldsymbol{x}_l$ (in the l-th iteration) are optimal in the sense that they minimise the so-called energy norm of the error vector. More precisely, if $\boldsymbol{x}^\star$ is a solution of the system $A\boldsymbol{x} = \boldsymbol{b}$, then $\boldsymbol{x}_l$ minimises $\|\boldsymbol{x}^\star - \boldsymbol{x}_l\|_A$ for the A-norm $\|\boldsymbol{y}\|_A := \sqrt{\boldsymbol{y}^\top A\boldsymbol{y}}$. More precisely, the CG method gives in the l-th iteration the best solution available in the generated subspace. Since the dimension of the Krylov subspace is increased in each iteration, theoretical convergence is achieved at latest after the n-th step of the method if the sought solution is in $\mathbb{R}^N$. In practice, numerical rounding errors occur and very large systems may suffer from severe convergence problems. Fortunately, this problem can be overcome by proper so-called preconditioning, which is a way to induce rapid convergence.

Appendix A.3. Preconditioned Conjugate Gradient Method

Preconditioning is a very complex topic with decades of research, for an excellent survey we refer the reader to [42]. When dealing with symmetric matrices, the incomplete Cholesky factorisation is mainly used to build a common and very efficient preconditioner for the CG method. Let us briefly describe the approach.

Incomplete Cholesky Factorisation

The complete decomposition of A will be given by $A = CC^\top + F$. If the lower triangular matrix C is allowed to have non-zero entries anywhere in the lower matrix, then F is the zero matrix and the decomposition is the standard Cholesky decomposition; however, if only the structure of entries in A is used to define C, then the factorisation will be incomplete. In other words, the lower triangular matrix C keeps the same non-zero pattern as that of the lower triangular part of A. The general form of the preconditioning then amounts to the transformation from $A\boldsymbol{x} = \boldsymbol{b}$ to $A^p\boldsymbol{x}^p = \boldsymbol{b}^p$ with

$$A^p = C^{-1}AC^{-\top},\ \boldsymbol{x}^p = C^\top \boldsymbol{x} \quad \text{and} \quad \boldsymbol{b}^p = C^{-1}\boldsymbol{b} \tag{A6}$$

The mentioned approach of using the same sparsity pattern of A for C is often called *no-fill* IC factorisation or IC(0). If one extends the sparsity pattern of C by additional non-zero elements then the closeness between the product $CC^\top$ and A is potentially improved. This proceeding is often denoted as numerical fill-in strategy IC(γ), where the parameter $\gamma > 0$ (called *drop tolerance*) describes a dropping criterion, cf. [2]. The approach can be described as follows: new fill-ins are only accepted if the absolute value of the elements is greater than the (local) drop tolerance γ. Adding fill-ins may obviously lead to a better preconditioner and a potentially better convergence rate. On the other hand, it becomes more computationally intensive to solve the underlying triangular systems and the preconditioner itself; therefore, a good selection of γ in the preconditioning method is essential.

Modified Incomplete Cholesky

When dealing with parabolic PDEs, the MIC factorisation can lead to an even better preconditioner, for an overview on MIC see [40,42,65]. The idea behind the modification is to force the preconditioner to have the same row sums as the original matrix A. This can be accomplished by adding the discarded fill-ins to the diagonal. The latter approach is known as MIC(0) and can be combined with the abovementioned drop tolerance strategy to MIC(γ). An essential requirement for the safe use of this technique is that A is a diagonally dominant M-matrix, which is the case here.

Overall, using PCG method leads to a potentially better convergence rate and speeds up the CG method dramatically, but a good choice of γ is crucial. In practice, cf. [41,42], good results are obtained for values of $\gamma \in [10^{-4}, 10^{-2}]$ but the optimal value naturally depends on the model problem; therefore, it will require a thorough study to identify the most useful parameter when using PCG for the GES application.

Appendix A.4. Model Order Reduction

MOR methods (also known as reduced order model methods) can be used to approximate the original high-dimensional linear and time-invariant first order ODE system (28), where the time invariance refers to the fact that the corresponding matrix is constant in time, by a very low-dimensional system, while preserving the main characteristics of the original system. Usually, MOR methods are defined by a projected model which rely on efficient numerical linear algebra techniques for problems involving large sparse dynamical systems. The underlying concept of projection methods is to approximate the high-dimensional state space vector $\boldsymbol{u}(t) \in \mathbb{R}^N$ by a reduced basis

$$\boldsymbol{u}(t) \approx V\boldsymbol{u}_r(t) \tag{A7}$$

with reduced vector $\boldsymbol{u}_r(t) \in \mathbb{R}^r$, projection matrix $V \in \mathbb{R}^{N\times r}$ and $r \ll N$. Applying this concept can be comprehended as projecting the original MICO dynamical system

$$\begin{cases} \dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K\boldsymbol{w}(t) \\ \boldsymbol{y}(t) = \boldsymbol{u}(t), \;\; \boldsymbol{u}(0) = \boldsymbol{u}^0 \end{cases} \tag{A8}$$

with the output variable $\boldsymbol{y}(t) \in \mathbb{R}^N$ and the initial condition $\boldsymbol{u}^0 \in \mathbb{R}^N$ into a reduced order model

$$\begin{cases} W^\top V\dot{\boldsymbol{u}}_r(t) = W^\top LV\boldsymbol{u}_r(t) + W^\top K\boldsymbol{w}(t) \\ \boldsymbol{y}_r(t) = V\boldsymbol{u}_r(t), \;\; \boldsymbol{u}_r(0) = \boldsymbol{u}^{r,0} \end{cases} \tag{A9}$$

by applying (A7) and left multiplication with the projection matrix $W^\top \in \mathbb{R}^{r\times N}$. By multiplication from left with $(W^\top V)^{-1}$, and assuming the inverse exists, the system (A9) leads to

$$\begin{cases} \dot{\boldsymbol{u}}_r(t) = (W^\top V)^{-1}W^\top LV\boldsymbol{u}_r(t) + (W^\top V)^{-1}W^\top K\boldsymbol{w}(t) \\ \boldsymbol{y}_r(t) = V\boldsymbol{u}_r(t), \;\; \boldsymbol{u}_r(0) = \boldsymbol{u}^{r,0} \end{cases} \tag{A10}$$

In order to avoid numerical problems the projection matrices are generally chosen as biorthonormal matrices $W^\top V = I$ so that the reduced system of order r has the following form

$$\begin{cases} \dot{\boldsymbol{u}}_r(t) = L_r\boldsymbol{u}_r(t) + K_r\boldsymbol{w}(t) \\ \boldsymbol{y}_r(t) = V\boldsymbol{u}_r(t), \;\; \boldsymbol{u}_r(0) = \boldsymbol{u}^{r,0} \end{cases} \tag{A11}$$

with $L_r = W^\top LV \in \mathbb{R}^{r\times r}$, $K_r = W^\top K \in \mathbb{R}^{r\times p}$, $\boldsymbol{y}_r(t) \in \mathbb{R}^N$ and $\boldsymbol{u}^{r,0} \in \mathbb{R}^r$. It should be noted that, within the formulation (A11), only the number of the state variables is reduced, whereas the number of inputs and outputs remains the same.

Frequently used techniques for constructing the projection matrices V and W are MCR, BT, POD and KSMOR. Attractive advantages of MCR and BT are that the reduced model inherits stability properties of the original system, moreover also (a priori) global error bounds may be guaranteed. In contrast, the POD concept based on data reduction makes this technique highly flexible, so it is often applied to nonlinear dynamical systems; however, the main drawback of MCR, BT and POD is the underlying approach for computing the projecting matrices, which is based on solving an eigenvalue problem, a Lyapunov equation or performing a SVD (sometimes in combination); therefore, the use of the three methods does not appear to be a practical for the underlying GES application. In contrast, the KSMOR approach is based on a series expansion of the system's transfer function and is nowadays the most frequently used MOR method for linear and large-scale dynamical systems.

Appendix A.5. Krylov Subspace Model Order Reduction

The main advantage of KSMOR is the fact that the projection method is based on Krylov subspaces, which yields high computational efficiency, and moreover often requires only a small dimension of the resulting reduced order model; however, in general, there is no guarantee for stability preserving during the reduction process, even if the original system is stable. Fortunately, the stability of the reduced order model can be guaranteed if a one-sided projection method, i.e., $W = V$, coupled with a negative semi-definite system matrix L is applied (see [66]), which is the case here.

The basic idea of KSMOR is the local approximation of the transfer function $\boldsymbol{H}(s)$ around a frequency σ of interest. In doing so, the transfer function is expanded into an infinite Taylor series in which the coefficients are defined as the moments. Based on this representation, the reduced order model is constructed so that some of the first moments of the original and reduced system are matched, which reflects why the approach is called moment matching.

In order to introduce the transfer function $\boldsymbol{H}(s)$, the Laplace transformation is applied to the full system (39), which results in

$$\begin{aligned} s\widetilde{\boldsymbol{U}}(s) - \widetilde{\boldsymbol{u}}(0) &= L\widetilde{\boldsymbol{U}}(s) + \widetilde{K}\widetilde{\boldsymbol{W}}(s) \\ \widetilde{\boldsymbol{Y}}(s) &= \widetilde{\boldsymbol{U}}(s) + \tfrac{\boldsymbol{u}^0}{s} \end{aligned} \tag{A12}$$

Assuming that the inverse $(sI - L)^{-1}$ exists, the latter corresponds to

$$\begin{aligned} \widetilde{\boldsymbol{U}}(s) &= (sI - L)^{-1}\widetilde{K}\widetilde{\boldsymbol{W}}(s) \\ \widetilde{\boldsymbol{Y}}(s) &= \widetilde{\boldsymbol{U}}(s) + \tfrac{\boldsymbol{u}^0}{s} = \left[(sI - L)^{-1}\widetilde{K}\right]\widetilde{\boldsymbol{W}}(s) + \tfrac{\boldsymbol{u}^0}{s} = \boldsymbol{H}(s)\widetilde{\boldsymbol{W}}(s) + \tfrac{\boldsymbol{u}^0}{s} \end{aligned} \tag{A13}$$

where the transfer function $\boldsymbol{H}(s) = (sI - L)^{-1}\widetilde{K}$ is expressed by the input $\widetilde{\boldsymbol{W}}(s)$ and the output $\widetilde{\boldsymbol{Y}}(s)$ in the frequency domain.

The aim of KSMOR is to construct a reduced dynamical system with transfer function $\widetilde{\boldsymbol{H}}(s)$, whose moments match the first q moments of the original transfer function $\boldsymbol{H}(s)$ as expressed in (41). Solving moment matching can be performed explicitly or implicitly. Explicit moment matching such as by Padé-type approximations are known to be numerically unstable, especially as the dimension of the reduced order model r grows; therefore, in practice, implicit moment matching based on Krylov subspaces is usually performed, which is a numerically stable method for calculating the moments. Proposed approaches to compute the Krylov subspaces are variations of the Arnoldi and the Lanczos process, see [5,6,67].

Block Arnoldi

For the construction of the Krylov subspace (42) the included inverse $(L - \sigma I)^{-1}$ leads anew to the task of solving large sparse systems of linear equations. This requires the application of sparse direct or sparse iterative solvers as introduced before. In this work, we focus on the MOR technique itself, therefore we apply the sparse direct solver as this gives the most accurate representation. Apart from that, for iterative solvers (only approximate solutions are generated), the moment matching property cannot hold in general, since the constructed subspace no longer matches the projection subspace, i.e.,

$$\operatorname{range}(V) \neq \mathcal{K}_q\left((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\widetilde{K}\right) \tag{A14}$$

We want to avoid this here as well.

The underlying construction of V has to deal with orthogonalisation processes. Typically, there can be a significant loss of orthogonality due to round-off errors as an orthogonalisation algorithm progresses. To resolve this problem and achieve a stable procedure that keeps the moment matching accuracy high, a reorthogonalisation of a new computed block with respect to all previous blocks is performed.

Remarks on the Reduced Solution

Based on the interior matching conditions and their associated calculations of fictitious values at the interfaces, the reduced solution $\widetilde{\boldsymbol{u}}_r$ within (43) must be re-enlarged in each time step into the original dimension by means of

$$\widetilde{\boldsymbol{u}} \approx V\widetilde{\boldsymbol{u}}_r \tag{A15}$$

Since the projection matrix V is not sparse the matrix-vector multiplication (A15) increases the computational costs significantly in each iteration. For this reason, we strive to avoid large-dimensional projection matrices for the GES application.

Appendix A.6. KSMOR⋆: Test Example

We now describe the modified KSMOR⋆ method at hand of a selected numerical experiment. In particular, we demonstrate the general viability of our approach, which makes it a candidate for efficiently solving large-scale dynamical systems with a large-dimensional input vector.

For the sake of completeness and to shed light on the technical differences between both methods, we analyse the quality of the solutions obtained through CN, KSMOR and KSMOR⋆ according to the L_2-error (48) recorded at a specific time t_F. To this end, we consider a two-dimensional linear heat equation on a rectangular domain $\Omega = \left[0, \frac{3}{2}\right] \times \left[0, 2\right]$ equipped with time- and space-dependent boundary conditions. The diffusion problem reads as

$$\partial_t u(x,y,t) = a\partial_{xx} u(x,y,t) + b\partial_{yy} u(x,y,t) + f(x,y), \quad t \in \left[0, t_F\right] \tag{A16}$$

where the fluxes and the source term are given by

$$a = \frac{1}{18\pi^2}, \quad b = \frac{1}{2\pi^2}, \quad f(x,y) = 2\cos\left(3\sqrt{2}\pi x\right)\cos\left(\sqrt{2}\pi y\right) \tag{A17}$$

The corresponding external boundary conditions are fixed to

$$\begin{aligned} g_1(y,t) &= u(0,y,t) = \cos\left(\sqrt{2}\pi y\right) \\ g_2(y,t) &= u\left(\tfrac{3}{2},y,t\right) = \cos\left(\pi y\right)e^{-t} + \cos\left(\tfrac{9\sqrt{2}}{2}\pi\right)\cos\left(\sqrt{2}\pi y\right) \\ h_1(x,t) &= u(x,0,t) = \sin\left(3\pi x\right)e^{-t} + \cos\left(3\sqrt{2}\pi x\right) \\ h_2(x,t) &= u(x,2,t) = \sin\left(3\pi x\right)e^{-t} + \cos\left(3\sqrt{2}\pi x\right)\cos\left(2\sqrt{2}\pi\right) \end{aligned} \tag{A18}$$

with the initial condition specified as

$$u(x,y,0) = \sin\left(3\pi x\right)\cos\left(\pi y\right) + \cos\left(3\sqrt{2}\pi x\right)\cos\left(\sqrt{2}\pi y\right) \tag{A19}$$

The resulting semi-discrete ODE system can be expressed via

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + \boldsymbol{g}_1 + K_{g_2}\boldsymbol{g}_2(t) + K_{h_1}\boldsymbol{h}_1(t) + K_{h_2}\boldsymbol{h}_2(t) + \boldsymbol{f}, \;\; \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{A20}$$

and its transformed system with zero initial conditions has the form

$$\dot{\widetilde{\boldsymbol{u}}}(t) = L\widetilde{\boldsymbol{u}}(t) + \widetilde{K}\widetilde{\boldsymbol{w}}(t), \quad \widetilde{\boldsymbol{u}}(0) = \boldsymbol{0} \tag{A21}$$

with

$$\widetilde{K} = \left[\boldsymbol{g}_1, K_{g_2}, K_{h_1}, K_{h_2}, \boldsymbol{f}, L\boldsymbol{u}^0\right], \quad \widetilde{\boldsymbol{w}}(t) = \left(1, \boldsymbol{g}_2(t), \boldsymbol{h}_1(t), \boldsymbol{h}_2(t), 1, 1\right)^\top \tag{A22}$$

Furthermore, we set $h = 0.02$, $\tau = 0.001$ and $t_F = 5$. In this case, the number of grid points amounts to $N = 7676$ and the individual input dimensions result in $\boldsymbol{g}_2(t) \in \mathbb{R}^{101}$ and $\boldsymbol{h}_1(t)$, $\boldsymbol{h}_2(t) \in \mathbb{R}^{76}$ with $K_{g_2} \in \mathbb{R}^{7676\times 101}$ and K_{h_1}, $K_{h_2} \in \mathbb{R}^{7676\times 76}$, respectively. In total, the input and the corresponding input matrix are of size $\widetilde{\boldsymbol{w}}(t) \in \mathbb{R}^{256}$ and $\widetilde{K} \in \mathbb{R}^{7676\times 256}$, respectively.

To compare the performances of KSMOR and KSMOR⋆, the result of the CN method is used as a reference solution. Let us specify that the computational costs (expressed in CPU time) of CN using sparse direct solver for the fixed parameters amount to 5.97 seconds. Obviously, the number of subspaces used within both KSMOR methods is still a free parameter and influences the approximation quality as well as the efficiency of the numerical schemes, see Figure A1. As expected, KSMOR leads to a reduction in the L_2-error as the number of subspaces is increased, but the computational costs are significantly higher compared to CN. Even if the subspace order $q = 1$ is used, the costs are almost twice as

high. Thus, this experiment clarifies the inefficiency of KSMOR for problems with a large number of inputs.

Conversely, the proposed KSMOR* method can be applied to tackle this problem. For applying this approach, snapshot data are required for $\boldsymbol{g}_2$, $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$, while the other inputs are constant and can be ignored. To construct the snapshot-based input matrix $\overline{K} \in \mathbb{R}^{7676\times(3s+3)}$ and the corresponding projection matrix $\overline{V}$, the model is simulated using the CN method for $\tau = \frac{t_F}{s}$ and the input data are stored as

$$W_1 = \left[\boldsymbol{g}_2(t_1), \ldots, \boldsymbol{g}_2(t_s)\right], \quad W_2 = \left[\boldsymbol{h}_1(t_1), \ldots, \boldsymbol{h}_1(t_s)\right], \quad W_3 = \left[\boldsymbol{h}_2(t_1), \ldots, \boldsymbol{h}_2(t_s)\right] \tag{A23}$$

with $W_1 \in \mathbb{R}^{101\times s}$ and W_2, $W_3 \in \mathbb{R}^{76\times s}$. The snapshot-based matrix is then determined via

$$\overline{K} = \left[\boldsymbol{g}_1, K_{g_2} W_1, K_{h_1} W_2, K_{h_2} W_3, \boldsymbol{f}, L\boldsymbol{u}^0\right] \tag{A24}$$

After the construction of $\overline{V}$ with $\sigma = 0$, the dimension reduction is applied to the original system (A21). For the sake of convenience, the latter snapshot dependence on s is abbreviated to KSMOR$^\star_s$. The results for KSMOR$^\star_2$, KSMOR$^\star_5$ and KSMOR$^\star_{10}$, shown in Figure A1 clearly demonstrate a much higher efficiency compared to CN and KSMOR. For instance, using KSMOR$^\star_5$ with $q = 10$ results in an excellent trade-off between quality and efficacy and can save approximately 95% and 80% of the computational time in relation to KSMOR and CN, respectively.

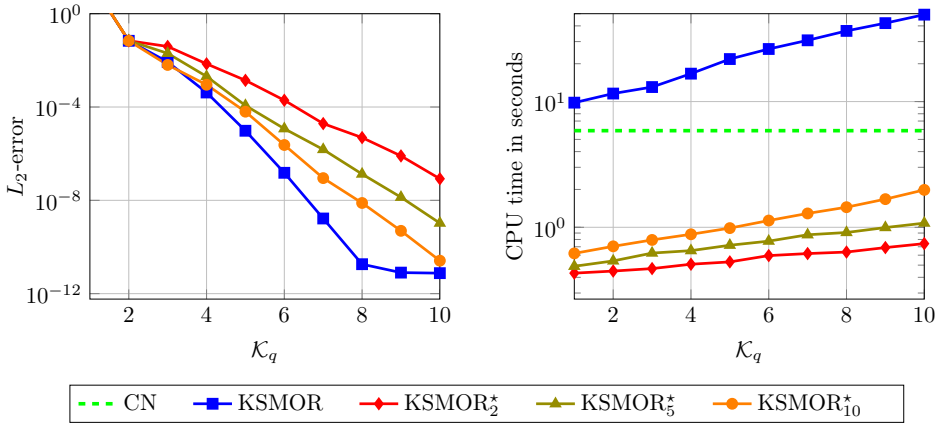


Figure A1. Results for the test example (A16)–(A19) at a fixed time $t_F = 5$. Comparison of the L_2-error (**left**) and the CPU time (**right**) between KSMOR and KSMOR* for a different number of Krylov subspaces $\mathcal{K}_q$. The reference solution is computed with the direct CN method. In experiments, we study different numbers of snapshots $s = \{2, 5, 10\}$ used for the KSMOR* method, which is abbreviated as KSMOR$^\star_s$. Obviously, the KSMOR* technique is much more efficient compared to the original KSMOR method as well as the CN method.

Appendix A.7. Summary of Numerical Solvers

Classic explicit methods are based on extremely simple evaluations as their computational costs corresponds to sparse matrix-vector multiplications. They are very easy to implement and due to their explicit nature well-suited for parallel computing such as GPUs. Unfortunately, semi-discretised ODE systems such as (28) are known to be stiff with respect to parabolic problems, meaning that explicit schemes suffer from severe time step size restrictions on τ to satisfy numerical stability; therefore, using a very small τ generally makes explicit methods unusable in practice.

In contrast, implicit methods may have no time step size restriction, but they require the inversion of large sparse matrices. This means that a very large, sparse system of equations must be solved at each time level, which can be computationally intensive and more difficult to parallelise. Fortunately, the underlying system considered here

is linear and can be efficiently solved by a factorisation or preconditioning technique that is precomputed once only. In particular, they are relatively simple to implement or rely on existing sophisticated software packages and have moderate computational and memory complexity. Consequently, implicit schemes are usually the best choice when considering linear parabolic problems; however, for applications with source terms and internal/external boundary conditions, it has to be kept in mind that the contributions of the sources must be updated at relatively small time intervals in order to obtain an accurate simulation. Thus, implicit methods have to solve many large sparse systems of linear equations, which leads to a high computational effort.

The accelerated explicit schemes FED and FSI are based on the decomposition of a box filter that can be factorised into a cycle of explicit linear diffusion steps. In doing so, the FED method uses stable and unstable time step sizes that violate the upper stability bound of the EE scheme significantly. To overcome some of the drawbacks of the basic FED method, the advanced version FSI can be used, which is based on a recurrence relation of the box filters. In this way, the FSI method ensures internal stability and is therefore of high practical use. In particular, both schemes are of explicit nature, well-suited for parallel computing and highly efficient with low memory demand, which makes them ideal for solving parabolic PDEs efficiently. They can easily be applied, assuming the underlying model problem has negative real eigenvalues. More precisely, stability and convergence are only theoretically guaranteed for symmetric negative semi-definite matrices, which is the case for the GES application; however, the proposed fast explicit schemes are not very suitable for extremely stiff parabolic problems, since extremely large eigenvalues require a very large number of inner cycle lengths n to achieve stability. Overall, FED and FSI are powerful solvers to efficiently solve parabolic problems with moderate stiffness, where the latter property usually holds when using a uniform and not too fine spatial discretization (as considered here).

MOR techniques are a popular tool for reducing computational complexity by approximating the large-scale dynamical system with a reduced, low-dimensional dynamical system while preserving the main characteristics of the original one. Nowadays, the KSMOR technique is the most widely used MOR method for solving large-scale semi-discrete ODE systems. Although moment matching methods are generally not stability preserving during the reduction process, the stability of the reduced order model can be guaranteed if a one-sided projection method, i.e., $V = W$, coupled with a negative semi-definite system matrix is applied (which is the case here). Furthermore, this technique, which is implicitly based on Krylov subspaces, often requires only a small dimension of the resulting reduced order model. Consequently, the numerically stable KSMOR method requires a relatively low computational effort and memory storage, which favours this technique for the reduction in large-scale dynamical systems. Although KSMOR is indeed a simple, powerful tool and has been extensively analysed over the past decades, there are still challenges within the reduction process, especially when solving model problems with a high number of inputs. To overcome this problem, we propose an adapted KSMOR* method that can solve systems with many inputs in a highly efficient way. The practicability is also described and tested using a test example.

Appendix B. A Synthetic Setup

In this experiment, we analyse the quality of the solutions obtained through the considered methods. To this end, we focus on the error measures at a fixed moment in time. We do not consider a full error computation along the temporal axis because our methods yield completely different sampling rates, potentially leading to unfair comparisons. In addition, we also specify the L_∞-error defined as

$$L_\infty = \max_{i=1,\dots,N} |\boldsymbol{u}_i - \widetilde{\boldsymbol{u}}_i| \tag{A25}$$

to analyse the performances of the methods.

We consider a simple linear heat equation on a two-dimensional square domain given by $\Omega = [0,1] \times [0,1]$ equipped with mixed homogeneous Dirichlet boundary conditions on left and right sides and homogeneous Neumann boundary conditions on top and bottom sides:

$$\begin{aligned} \partial_t u(x,y,t) &= \kappa\big(\partial_{xx} u(x,y,t) + \partial_{yy} u(x,y,t)\big), \quad (x,y,t) \in \Omega \times [0,t_F] \\ u(0,y,t) &= u(1,y,t) = 0, \quad \partial_y u(x,0,t) = \partial_y u(x,1,t) = 0 \\ u(x,y,0) &= x(1-x)y \end{aligned} \tag{A26}$$

The problem (A26) can be solved analytically using the method of separation of variables and its solution is given by

$$\begin{aligned} u(x,y,t) = 2 \sum_{n=1}^{\infty} \tfrac{(1-(-1)^n)}{\pi^3 n^3} e^{-\kappa \pi^2 n^2 t} \sin(n\pi x) \\ - \frac{16}{\pi^5} \sum_{n=1}^{\infty} \sum_{m=1}^{\infty} \tfrac{((-1)^n-1)((-1)^m-1)}{n^3 m^2} e^{-\kappa \pi^2 \left(n^2+m^2\right) t} \sin(n\pi x) \cos(m\pi y) \end{aligned} \tag{A27}$$

Considering these boundary conditions results in $K \equiv 0$, i.e., no input vector $\boldsymbol{w}(t)$ exists, and therefore the corresponding semi-discrete problem reads as

$$\begin{cases} \dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \boldsymbol{u}(t), \;\; \boldsymbol{u}(0) = \boldsymbol{u}^0 \end{cases} \tag{A28}$$

This implies that the original KSMOR method can be used here. Moreover, the Laplacian matrix L is negative definite. The still free model parameters are fixed to $t_F = 1$, $\kappa = 1$ and $h = 0.01$ ($N = 10201$ grid points). Let us note that for comparison with the ground truth we took great care to evaluate the analytical solution in power series form as accurately as possible. To this end, we truncated the summation at $n = m = 10$. The results of the numerical evaluation for varying cycle numbers M and time step sizes τ are presented in Table A1.

Results on EE: For the numerical solution of the artificial model problem via the EE method, the knowledge of the upper bound $\tau_{\max}$ is necessary. According to Gershgorin's circle theorem the time step size restriction is given by $\tau_{\max} = 2.5 \times 10^{-5}$ seconds. Consequently, the application of EE requires performing 40,000 iterations and leads to a simulation time of around 3.6 seconds. A reduction in $\tau_{\max}$ does not give better results; compare Table A1.

Results on FED and FSI: The evaluation of the explicit schemes FED and FSI clearly demonstrate their numerical properties and performances, mentioned in Section 4.1.1. Both methods achieve exactly the same output; however, the FSI method is more efficient.

Table A1. Evaluation of the accuracy of the suggested methods at $t_F = 1$ on the artificial setup from (A26). The analytical solution (A27) is evaluated with $\kappa = 1$ and the summation over n and m is truncated at $n = m = 10$. The grid size is fixed at $h = 0.01$. The stopping criterion of the CG method is based on the Euclidean norm of the relative residual with $\varepsilon = 10^{-4}$. Moreover, KSMOR is performed for $\mathcal{K}_4$ and $\sigma = 0$, whereby $q = 4$ is sufficient to achieve the same results than the direct CN method.

Method	τ or M	L_∞	L_2	CPU(s)
EE	$\tau_{\max}$	3.02×10^{-6}	2.22×10^{-4}	3.58
	$\tau_{\max}/10$	3.03×10^{-6}	2.23×10^{-4}	41.53
FED	$M = 10$	6.18×10^{-6}	4.39×10^{-4}	0.19
	$M = 100$	1.38×10^{-6}	1.04×10^{-4}	0.59
	$M = 1000$	2.85×10^{-6}	2.10×10^{-4}	1.87
	$M = 5000$	2.99×10^{-6}	2.20×10^{-4}	4.22

Table A1. *Cont.*

Method	τ or M	L_∞	L_2	CPU(s)
FSI	$M = 10$	6.18×10^{-6}	4.39×10^{-4}	0.13
	$M = 100$	1.38×10^{-6}	1.04×10^{-4}	0.42
	$M = 1000$	2.85×10^{-6}	2.10×10^{-4}	1.32
	$M = 5000$	2.99×10^{-6}	2.20×10^{-4}	2.95
Direct CN	$\tau = 2.5 \times 10^{-2}$	3.70×10^{-3}	3.31×10^{-2}	0.08
	$\tau = 2.5 \times 10^{-3}$	3.03×10^{-6}	2.22×10^{-4}	0.61
	$\tau = 2.5 \times 10^{-4}$	3.03×10^{-6}	2.23×10^{-4}	5.91
Iterative CN	$\tau = 2.5 \times 10^{-2}$	3.70×10^{-3}	3.31×10^{-2}	0.61
	$\tau = 2.5 \times 10^{-3}$	3.03×10^{-6}	2.22×10^{-4}	1.18
	$\tau = 2.5 \times 10^{-4}$	3.01×10^{-6}	2.21×10^{-4}	3.57
KSMOR	$\tau = 2.5 \times 10^{-2}$	3.70×10^{-3}	3.31×10^{-2}	0.03
	$\tau = 2.5 \times 10^{-3}$	3.03×10^{-6}	2.22×10^{-4}	0.04
	$\tau = 2.5 \times 10^{-4}$	3.03×10^{-6}	2.23×10^{-4}	0.06

Results on Direct and Iterative CN: As expected, the CN method substantially outperforms the EE method. Using $\tau = 2.5 \times 10^{-3}$ produces results with same accuracy, but in significantly less CPU time. Furthermore, it is obvious that the direct solver outperforms the iterative solver for $\varepsilon = 10^{-4}$. The iterative CN is only more efficient when using small time increments, which is a consequence of a faster convergence rate due to the small condition number. Note that due to the small size of the linear system, preconditioning is not conducive and the iterative CN method is performed employing the CG scheme. In addition, Table A2 shows a comparison for different values of ε. Using $\varepsilon = \{10^{-6}, 10^{-5}\}$ as stopping criterion generates the same accuracy results as the direct solver.

Results on KSMOR: As mentioned in Section 4.3.1, the system (A28) is firstly transformed into a system with zero initial conditions. Based on the matrix definiteness, the expansion point is fixed with $\sigma = 0$ and the projection subspace $V = \mathcal{K}_q\left(L^{-1}, L^{-1}L\boldsymbol{u}^0\right)$ is generated using the Arnoldi method. In this experiment we observe that matching the first $q = 4$ moments works optimally. In this case, the KSMOR method computes extremely fast approximations and generates exactly the same results as the original full CN method.

Outcome: Obviously, the EE scheme is the most accurate method, whereby the approximations of all other methods converge towards the EE result. In total, the KSMOR technique is incredibly computationally fast with the same accuracy compared to the direct CN scheme, thus demonstrating its superior performance.

Table A2. Comparison of the iterative CN scheme linked to the CG method in terms of the termination of the algorithm using the relative residual for different values of ε. We evaluate the accuracy of the artificial setup (A26) at $t_F = 1$ for the grid size $h = 0.01$. The analytical solution (A27) is evaluated for $k = 1$ and truncated at $n = m = 10$.

Iterative CN	ε	L_∞	L_2	CG-iter	CPU(s)
$\tau = 2.5 \times 10^{-4}$	$\varepsilon = 10^{-6}$	3.03×10^{-6}	2.23×10^{-4}	14342	4.35
	$\varepsilon = 10^{-5}$	3.03×10^{-6}	2.23×10^{-4}	12470	4.11
	$\varepsilon = 10^{-4}$	3.01×10^{-6}	2.21×10^{-4}	10653	3.51
	$\varepsilon = 10^{-3}$	2.13×10^{-6}	1.56×10^{-4}	9224	3.25
	$\varepsilon = 10^{-2}$	1.82×10^{-6}	1.33×10^{-4}	9274	3.31
	$\varepsilon = 10^{-1}$	6.46×10^{-6}	4.59×10^{-4}	9262	3.30

Appendix C. Geothermal Energy Storage Simulation without Source

Appendix C.1. Results on Direct CN

We examine the performance using the sparse direct solver applied to IE and CN. The results illustrated in Figure A2 demonstrate the expected superiority of the CN scheme due to its second-order accuracy in time. For this reason, the IE method is not considered in this work for the GES application.

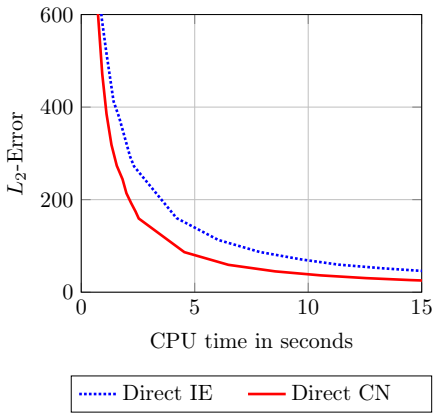


Figure A2. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and corresponding CPU time between direct IE and direct CN at $t_F = 2.609 \times 10^6$ for $h = 0.04$. The reference solution is computed using EE. The CN scheme clearly outperforms the IE method.

Appendix C.2. Results on KSMOR

The performance of the original KSMOR technique for $q = \{2, 3, 5\}$ is shown in Figure A3. The reduction technique provides very reasonable results, but is also linked to extremely high computational costs. This is hardly surprising as a common weakness of KSMOR is the handling of large-scale input vectors.

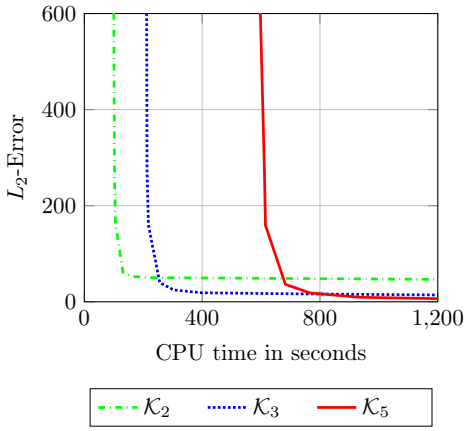


Figure A3. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time using KSMOR at $t_F = 2.609 \times 10^6$ for $h = 0.04$. KSMOR is used by constructing subspaces with $\widetilde{\boldsymbol{w}}(t) \in \mathbb{R}^{1106}$, $\widetilde{K} \in \mathbb{R}^{N \times 1106}$ for $q = \{2, 3, 5\}$. The reference solution is computed using EE. The results are accurate, but also linked to extremely high computational costs.

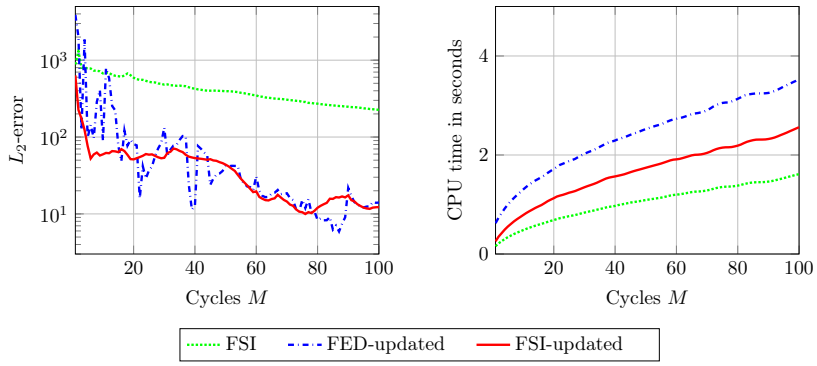


Figure A4. Results of the two-dimensional GES experiment without source: comparison of the computed L_2-error (**left**) and the corresponding CPU time (**right**) between FSI, FED-updated and FSI-updated for a different number of cycles M at $t_F = 2.609 \times 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. To ensure numerical stability within the FED method we apply a Leja-ordering with $\kappa = 7$, for more details see [17]. Inner updates significantly improve the performance of both schemes; however, FED-updated may lead to highly unstable intermediate solutions and therefore also affects the final numerical solution, especially if a small number of cycles M is used. Contrary, FSI-updated is absolutely stable and causes lower computational costs than FED-updated.

Appendix C.3. Results on FED and FSI

We experimentally confirm the considerations mentioned in Section 4.1.1. For the realisation, the differences between the basic scheme and the adapted versions using inner updates of $w^{m,k}$ are examined in Figure A4. In case of the basic scheme, we only consider the FSI technique, since both methods yield equally numerical solutions. Let us first note that FED and FSI using inner updates cannot provide the same output any more because the schemes are built on different integration points. For convenience only we denote the adapted versions here as *FED-updated* and *FSI-updated*. As expected, FED-updated is highly unstable, especially for a small number of cycles M. The latter fact is due to unstable intermediate solutions as a consequence of a rearrangement, where we used the Leja-ordering with $\kappa = 7$ as suggested in [17]. This implies the use of a relatively large cycle Number M, nevertheless, a stable solution is not guaranteed. In contrast, FSI-updated provides stable solutions and is additionally much more efficient.

Using the basic FSI scheme is less computationally intensive than using FSI-updated, at the cost of a significant loss of performance. The lack of efficiency between the two FSI versions is also presented in Figure A5.

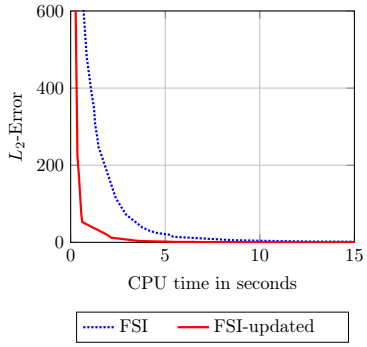


Figure A5. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time between FSI and FSI-updated at $t_F = 2.609 \times 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. The scheme FSI-updated clearly outperforms their counterpart without inner updates.

Appendix C.4. Results on a Finer Grid

The performances of the proposed methods should also be taken into consideration on a finer grid. Especially in case of explicit methods, where the involving stability requirement depends quadratically on the spatial grid size. Moreover, the computational costs of solving linear equations belonging to very large systems are also of interest.

Therefore, we rerun the experiment for $h = 0.01$, which corresponds to $N = 1,502,501$ grid points. The upper bound is given by $\tau_{\max} = 26.09$ and applying EE requires 100,000 iterations, which leads to a simulation time of around 2800 s. The final outcome, illustrated in Figure A6, gives exactly the same rating as before, again, the FSI method achieves significantly higher efficiency.

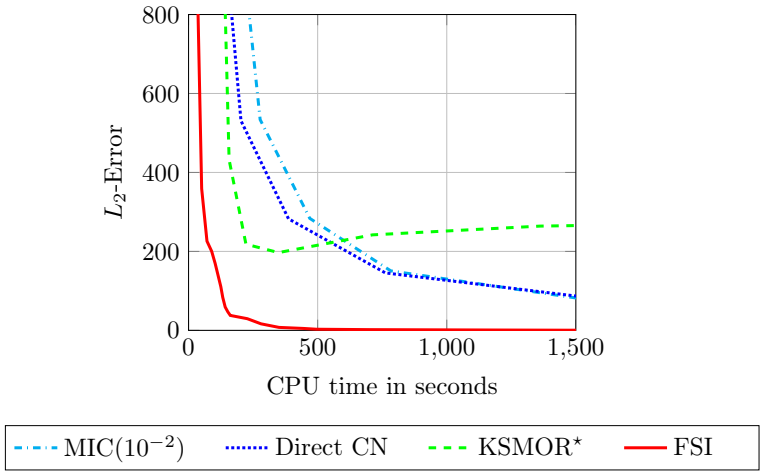


Figure A6. Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time between MIC(10^{-2}), direct CN, KSMOR* and FSI at $t_F = 2.609 \times 10^6$ for $h = 0.01$ ($N = 1,502,501$ grid points). KSMOR* is performed with $q = 3$ and $s = 20$. The reference solution is computed with the EE method. The FSI method clearly outperforms all other methods even for finer grids.

Appendix D. Geothermal Energy Storage Simulation on Real Data

Appendix D.1. Initial Heat Distribution

The initial heat distribution shown in Figure A7 builds upon given temperature probes and is generated by Laplace interpolation as described in Section 2.4. For a more precise start initialisation via (10), the model problem (11) is equipped with suitable boundary conditions.

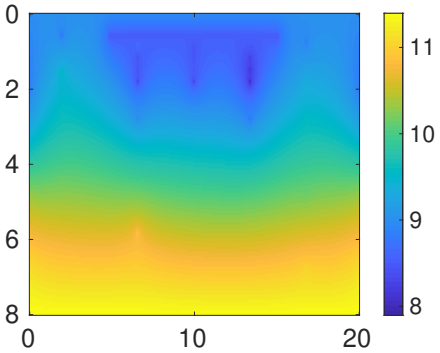


Figure A7. The initial heat distribution computed via Laplace interpolation using the initial temperature given by temperature probes at time $t = 0$, see Section 2.4. The interpolation task is performed with suitable boundary conditions.

Appendix D.2. Results for Temperature Probe B1

In addition to the results previously shown in Figures 11 and 12, we give in Table A3 a further evaluation with respect to temperature probe B1 (cf. middle column) for varying cycle numbers M and time step sizes τ, which is useful to better classify the accuracy of the different methods with the findings described in this work. In this context, the L_∞- and L_2-errors regarding the entire simulation domain (cf. right column) at the stopping time t_F are also compared. A deeper analysis of Table A3 clearly shows that FSI is the most efficient method, whereas direct CN provides the worst performance. The KSMOR* method also provides fast and accurate local solutions (away from interfaces), as indicated by the L_∞- and L_2-errors along the temporal axis shown in the *middle* column. Conversely, as already mentioned, accurate global solutions are provided only for a reasonably high number of snapshots and Krylov subspaces. The latter aspect is observable from the error measurements for the entire simulation domain in the *right* column, more precisely, the values for the L_∞- and L_2-errors do not converge toward the reference solution when KSMOR* with $q = 2,\ s = 10$ is used.

Table A3. Results of the GES experiment with real data: evaluation of the accuracy and the corresponding CPU time between FSI, direct CN and KSMOR* with $q = 2,\ s = 10$. The parameters are fixed to $t_F = 21 \times 10^6$ and $h = 0.04$ ($N = 100,701$ grid points). Special care should be taken because the errors presented with respect to L_∞ and L_2 are evaluated in two ways: errors evaluated along the temporal axis (**middle**) and at the stopping time t_F (**right**) for temperature probe B1 (3rd from the top) and the entire simulation domain, respectively. The reference solution is computed with EE.

Method	τ or M	**CPU(s)**	L_∞	L_2	L_∞	L_2
FSI	$M = 10$	2.17	5.16	963.39	7.23	279.33
	$M = 50$	4.57	1.42	219.99	1.78	85.98
	$M = 100$	6.53	1.03	99.51	0.41	23.18
	$M = 400$	12.73	0.33	35.95	0.27	8,97
	$M = 1000$	19.89	0.09	8.01	0.16	4.47
	$M = 4000$	40.81	0.02	1.48	0.03	0.75
Direct CN	$\tau = 2.1082 \times 10^6$	0.53	13.75	4141.33	6.51	538.29
	$\tau = 4.2163 \times 10^5$	1.33	7.96	2463.05	4.22	311.91
	$\tau = 1.0541 \times 10^5$	4.34	4.16	1165.35	1.05	100.59
	$\tau = 35136$	12.39	1.69	465.06	0.44	52.09
	$\tau = 21082$	20.29	1.06	270.81	0.34	26.88
	$\tau = 10800$	40.61	0.56	147.87	0.18	18.28
KSMOR*	$\tau = 2.1082 \times 10^6$	2.16	13.74	4138.84	6.51	542.28
	$\tau = 1.0541 \times 10^5$	2.94	4.16	1154.36	1.51	100.81
	$\tau = 21082$	6.22	1.09	256.36	0.91	38.07
	$\tau = 6912$	14.48	0.39	82.49	0.97	33.01
	$\tau = 3904$	22.58	0.24	47.57	0.98	32.50
	$\tau = 2196$	42.95	0.17	34.71	0.99	32.65

References

1. Davis, T.A. *Direct Methods for Sparse Linear Systems*; SIAM: Philadelphia, PA, USA, 2006.
2. Saad, Y. *Iterative Methods For Sparse Linear Systems*, 2nd ed.; SIAM: Philadelphia, PA, USA, 2003.
3. Grewenig, S.; Weickert, J.; Bruhn, A. From Box Filtering to Fast Explicit Diffusion. In Proceedings of the Pattern Recognition, Darmstadt, Germany, 22–24 September 2010; Goesele, M., Roth, S., Kuijper, A., Schiele, B., Schindler, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6376, pp. 533–542.
4. Hafner, D.; Ochs, P.; Weickert, J.; Reißel, M.; Grewenig, S. FSI Schemes: Fast Semi-Iterative Solvers for PDEs and Optimisation Methods. In Proceedings of the Pattern Recognition, Cancun, Mexico, 4–8 December 2016; Rosenhahn, B., Andres, B., Eds.; Springer: Cham, Switzerland, 2016; Volume 9796, pp. 91–102.
5. Antoulas, A.C. *Approximation of Large-Scale Dynamical Systems*; SIAM: Philadelphia, PA, USA, 2005.

6. Antoulas, A.C.; Ionutiu, R.; Martins, N.; ter Maten, E.J.W.; Mohaghegh, K.; Pulch, R.; Rommes, J.; Saadvandi, M.; Striebel, M. Model Order Reduction: Methods, Concepts and Properties. In *Coupled Multiscale Simulation and Optimization in Nanoelectronics*; Günther, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2015; pp. 159–265.
7. Šindler, J.; Suleng, A.; Olsen, T.J.; Bárta, P. Krylov Model Order Reduction of a Thermal Subsea Model. *Int. J. Mech. Aerospace Ind. Mechatron. Manuf. Eng.* **2013**, *7*, 842–849.
8. He, X.; Kong, Q.; Xiao, Z. Fast Simulation Methods for Dynamic Heat Transfer through Building Envelope Based on Model-Order-Reduction. *Procedia Eng.* **2015**, *121*, 1764–1771.
9. Rother, S.; Beitelschmidt, M. Load Snapshot Decomposition to Consider Heat Radiation in Thermal Model Order Reduction. *IFAC PapersOnLine* **2018**, *51*, 667–672.
10. Benner, P.; Herzog, R.; Lang, N.; Riedel, I.; Saak, J. Comparison of model order reduction methods for optimal sensor placement for thermo-elastic models. *Eng. Optim.* **2019**, *51*, 465–483.
11. Douglas, J. On the Numerical Integration $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial u}{\partial t}$ by Implicit Methods. *J. Soc. Ind. Appl. Math.* **1955**, *3*, 42–65.
12. Peaceman, D.W.; Rachford, J.H.H. The Numerical Solution of Parabolic and Elliptic Differential Equations. *J. Soc. Ind. Appl. Math.* **1955**, *3*, 28–41.
13. Weickert, J.; Zuiderveld, K.J.; ter Haar Romeny, B.M.; Niessen, W.J. Parallel Implementations of AOS Schemes: A Fast Way of Nonlinear Diffusion Filtering. In Proceedings of the International Conference on Image Processing, Santa Barbara, CA, USA, 26–28 October 1997; Volume 3, pp. 396–399.
14. Weickert, J.; ter Haar Romeny, B.M.; Viergever, M.A. Efficient and Reliable Schemes for Nonlinear Diffusion Filtering. *IEEE Trans. Image Process.* **1998**. *7*, 398–410.
15. Barash, D.; Israeli, M.; Kimmel, R. An Accurate Operator Splitting Scheme for Nonlinear Difusion Filtering. In Proceedings of the Scale-Space and Morphology in Computer Vision, Vancouver, BC, Canada, 7–8 July 2001; Kerckhove, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2106, pp. 281–289.
16. Bähr, M.; Breuß, M.; Wunderlich, R. Fast explicit diffiusion for long-time integration of parabolic problems. *AIP Conf. Proc.* **2017**, *1863*, 410002.
17. Grewenig, S. Fast Explicit Methods for PDE-Based Image Analysis. Ph.D. Thesis, Department of Mathematics, Saarland University, Saarbrücken, Germany, 2013.
18. Bähr, M. Efficient Time Integration Methods for Linear Parabolic Partial Differential Equations with Applications. Ph.D. Thesis, BTU Cottbus—Senftenberg, Senftenberg, Germany, 2022. https://doi.org/10.26127/BTUOpen-5753.
19. Shen, L.S.; Ramsey, J.W. An investigation of transient, two-dimensional coupled heat and moisture flow in the soil surrounding a basement wall. *Int. J. Heat Mass Transf.* **1988**, *31*, 1517–1527.
20. Takam, P.; Wunderlich, R.; Pamen, O. Short-Term Behavior of a Geothermal Energy Storage: Modeling and Theoretical Results. *arXiv* **2021**, arXiv:2104.05005.
21. Baehr, H.D.; Stephan, K. *Heat and Mass Transfer*, 3rd ed.; Springer: Berlin/Heidelberg, Germany, 2011.
22. Galić, I.; Weickert, J.; Welk, M.; Bruhn, A.; Belyaev, A.; Seidel, H.P. Towards PDE-Based Image Compression. In Proceedings of the Variational, Geometric, and Level Set Methods in Computer Vision, Beijing, China, 16 October 2005; Paragios, N., Faugeras, O., Chan, T., Schnörr, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3752, pp. 37–48.
23. Schönlieb, C.B. *Partial Differential Equation Methods for Image Inpainting*, 1st ed.; Cambridge University Press: Cambridge, UK, 2015.
24. Hickson, R.I.; Barry, S.I.; Mercer, G.N.; Sidhu, H.S. Finite difference schemes for multilayer diffusion. *Math. Comput. Model.* **2011**, *54*, 210–220.
25. Gerschgorin, S.A. Über die Abgrenzung der Eigenwerte einer Matrix. *Izv. Akad. Nauk SSSR Ser. Mat.* **1931**, *7*, 749–754.
26. Hairer, E.; Nørsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 1993.
27. Hairer, E.; Wanner, G. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 1996.
28. Smith, G.D. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, 3rd ed.; Clarendon Press: Oxford, UK, 1985.
29. Weickert, J.; Grewenig, S.; Schroers, C.; Bruhn, A. Cyclic Schemes for PDE-Based Image Analysis. *Int. J. Comput. Vis.* **2016**, *118*, 275–299.
30. Gentzsch, M.; Schlüter, A. Über ein Einschrittverfahren mit zyklischer Schrittweitenänderung zur Lösung parabolischer Differentialgleichungen. *Z. Angew. Math. Mech.* **1978**, *58*, T415–T416.
31. Gwosdek, P.; Zimmer, H.L.; Grewenig, S.; Bruhn, A.; Weickert, J. A Highly Efficient GPU Implementation for Variational Optic Flow Based on the Euler-Lagrange Framework. In Proceedings of the Trends and Topics in Computer Vision, Crete, Greece, 10–11 September 2010; Kutulakos, K.N., Ed. Springer: Berlin/Heidelberg, Germany, 2012, Volume 6554, pp. 372–383.
32. Tomasson, J.; Ochs, P.; Weickert, J. AFSI: Adaptive restart for fast semi-iterative schemes for convex optimisation. In Proceedings of the German Conference on Pattern Recognition (GCPR), Stuttgart, Germany, 9–12 October 2018; Brox, T., Bruhn, A., Fritz, M., Eds.; Springer: Cham, Switzerland, 2019; Volume 11269, pp. 669–681.
33. Jumakulyyev, I.; Schultz, T. Fourth-Order Anisotropic Diffusion for Inpainting and Image Compression. In *Anisotropy Across Fields and Scales*; Springer: Cham, Switzerland, 2021; pp. 99–124.

34. Alt, T.; Schrader, K.; Augustin, M.; Peter, P.; Weickert, J. Connections between Numerical Algorithms for PDEs and Neural Networks. *arXiv* **2021**, arXiv:2107.14742. https://doi.org/10.48550/ARXIV.2107.14742.
35. Bähr, M.; Dachsel, R.; Breuß, M. Fast Solvers for Solving Shape Matching by Time Integration. In Proceedings of the OAGM Workshop 2018, Tyrol, Austria, 15–16 May 2018; Welk, M.; Urschler, M.; Roth, P.M., Eds.; Verlag der TU Graz: Graz, Austria, 2018; pp. 65–72.
36. Davis, T.A.; Rajamanickam, S.; Sid-Lakhdar, W.M. A survey of direct methods for sparse linear systems. *Acta Numer.* **2016**, *25*, 383–566.
37. Barrett, R.; Berry, M.; Chan, T.F.; Demmel, J.; Donato, J.; Dongarra, J.; Eijkhout, V.; Pozo, R.; Romine, C.; van der Vorst, H. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*; SIAM: Philadelphia, PA, USA, 1994.
38. Kelley, C.T. *Iterative Methods for Linear and Nonlinear Equations*; SIAM: Philadelphia, PA, USA, 1995.
39. Hestenes, M.R.; Stiefel, E. Methods of Conjugate Gradients for Solving Linear Systems. *J. Res. Natl. Bur. Stand.* **1952**, *6*, 409–436.
40. Meurant, G.A. *Computer Solution of Large Linear Systems*, 1st ed.; Elsevier: Amsterdam, The Netherlands, 1999.
41. Bähr, M.; Breuß, M.; Quéau, Y.; Boroujerdi, A.S.; Durou, J.D. Fast and accurate surface normal integration on non-rectangular domains. *Comput. Vis. Media* **2017**, *3*, 107–129.
42. Benzi, M. Preconditioning Techniques for Large Linear Systems: A Survey. *J. Comput. Phys.* **2002**, *182*, 418–477.
43. Trottenberg, U.; Oosterlee, C.W.; Schüller, A. *Multigrid*, 1st ed.; Academic Press: Cambridge, MA, USA, 2000.
44. Tatebe, O. The Multigrid Preconditioned Conjugate Gradient Method. In Proceedings of the Sixth Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, USA, 4–9 April 1993; Melson, N.D., McCormick, S.F., Manteuffel, T.A., Eds.; NASA: Washington, DC, USA, 1993, pp. 621–634.
45. Krishnan, D.; Fattal, R.; Szeliski, R. Efficient Preconditioning of Laplacian Matrices for Computer Graphics. *ACM Trans. Graph.* **2013**, *32*, 142.
46. Antoulas, A.C.; Sorensen, D.C.; Gugercin, S. A survey of model reduction methods for large-scale systems. In *Structured Matrices in Mathematics, Computer Science, and Engineering I*; American Mathematical Society: Providence, RI, USA, 2001.
47. Baur, U.; Benner, P.; Feng, L. Model Order Reduction for Linear and Nonlinear Systems: A System-Theoretic Perspective. *Arch. Comput. Methods Eng.* **2014**, *21*, 331–358.
48. Falkiewicz, N.J.; Cesnik, C.E.S. Proper Orthogonal Decomposition for Reduced-Order Thermal Solution in Hypersonic Aerothermoelastic Simulations. *AIAA J.* **2011**, *49*, 994–1009.
49. Ojo, S.O.; Grivet-Talocia, S.; Paggi, M. Model order reduction applied to heat conduction in photovoltaic modules. *Compos. Struct.* **2015**, *119*, 477–486.
50. Grimme, E.J. Krylov Projection Methods for Model Reduction. Ph.D. Thesis, Department of Electrical Engineering, University of Illinois at Urbana-Champaign, Champaign, IL, USA, 1997.
51. Freund, R.W. Krylov-subspace methods for reduced-order modeling in circuit simulation. *J. Comput. Appl. Math.* **2000**, *123*, 395–421.
52. Bai, Z. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Appl. Numer. Math.* **2002**, *43*, 9–44.
53. Antoulas, A.C.; Beattie, C.A.; Gugercin, S. Interpolatory Model Reduction of Large-Scale Dynamical Systems. In *Efficient Modeling and Control of Large-Scale Systems*; Mohammadpour, J., Grigoriadis, K.M., Eds.; Springer: Boston, MA, USA, 2010; pp. 3–58.
54. Feng, L.H.; Koziol, D.; Rudnyi, E.B.; Korvink, J.G. Model Order Reduction for Scanning Electrochemical Microscope: The treatment of nonzero initial condition. *Proc. IEEE Sens.* **2004**, *3*, 1236–1239.
55. Lin, Y.; Bao, L.; Wei, Y. Order reduction of bilinear MIMO dynamical systems using new block Krylov subspaces. *Comput. Math. Appl.* **2009**, *58*, 1093–1102.
56. Feldmann, P.; Liu, F. Sparse and efficient reduced order modeling of linear subcircuits with large number of terminals. In Proceedings of the IEEE/ACM International Conference on Computer—Aided Design, San Jose, CA, USA, 7–11 November 2004; pp. 88–92.
57. Feldmann, P. Model Order Reduction Techniques for Linear Systems with Large Numbers of Terminals. In Proceedings of the Conference on Design, Automation and Test in Europe, Paris, France, 16–20 February 2004; p. 20944.
58. Li, P.; Shi, W. Model Order Reduction of Linear Networks with Massive Ports via Frequency-Dependent Port Packing. In Proceedings of the the 43rd ACM/IEEE Annual Design Automation Conference, San Francisco, CA, USA, 24–28 July 2006; pp. 267–272.
59. Benner, P.; Feng, L.; Rudnyi, E.B. Using the Superposition Property for Model Reduction of Linear Systems with a Large Number of Inputs. In Proceedings of the 18th International Symposium on Mathematical Theory of Networks & Systems, Blacksburg, VA, USA, 28 July–1 August 2008.
60. Benner, P.; Schneider, A. Model Reduction for Linear Descriptor Systems with Many Ports. In Proceedings of the Progress in Industrial Mathematics at ECMI 2010; Günther, M., Bartel, A., Brunk, M., Schöps, S., Striebel, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 137–143.
61. Jadhav, O.; Rudnyi, E.B.; Bechthold, T. Load Snapshot Based Nonlinear-Input Model Order Reduction of a Thermal Human Tissue Model. In Proceedings of the 12th International Conference on Scientific Computing in Electrical Engineering, Sicily, Italy, 5–9 September 2018.
62. Benner, P.; Schneider, A. Reduced Representation of Power Grid Models. In *System Reduction for Nanoscale IC Design*; Benner, P., Ed.; Springer International Publishing: Cham, Switzerland, 2017; pp. 87–134.

63. Evans, L.C. *Partial Differential Equations*, 2nd ed.; American Mathematical Society: Providence, RI, USA, 2010.
64. Mang, A.; Toma, A.; Schütz, T.A.; Becker, S.; Buzug, T.M.; Eckey, T.; Mohr, C.; Petersen, D. Biophysical modeling of brain tumor progression: From unconditionally stable explicit time integration to an inverse problem with parabolic PDE constraints for model calibration. *Med. Phys.* **2012**, *39*, 4444–4459.
65. Gustafsson, I. A class of first order factorization methods. *BIT Numer. Math.* **1978**, *18*, 142–156.
66. Selga, R.C.; Lohmann, B.; Eid, R. Stability Preservation in Projection-based Model Order Reduction of Large Scale Systems. *Eur. J. Control* **2012**, *18*, 122–32.
67. Lohmann, B.; Salimbahrami, B. *Introduction to Krylov Subspace Methods in Model Order Reduction*; Technical Report; Institute of Automation, University of Bremen: Bremen, Germany, 2003.