CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2010; **00**:1–21
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

Variability of Stencil Computations for Porous Media

A. Grebhahn[1]*, C. Engwer[2], M. Bolten[3], S. Apel[1]

[1]*Department of Informatics and Mathematics, University of Passau, Germany*
[2]*Institute for Computational and Applied Mathematics, Westfälische Wilhelms-Universität Münster, Germany*
[3]*Institute of Mathematics, University of Kassel, Germany*

SUMMARY

Many problems formulated in terms of partial differential equations lead to stencil-type structures after applying an appropriate structured discretization. On the one hand, exploiting these stencil structures in simulations can lead to massive performance improvements, compared to forming a sparse matrix. On the other hand, the generality of the simulation is restricted, depending on the exact definition of the stencils. In this article, we discuss the variability of stencils in the domain of porous-media applications and present a family of models that grows in complexity. To demonstrate the relation between equation and discretization on the resulting stencil used to simulate the equation, we consider four models from the porous media domain. This way, we describe the influence of design decisions made during the discretization on the shape of stencils, to give applications engineers information on the variability they have to consider. This leads us to two variability models that shall help application engineers to understand the complexity and choices of stencil computations in the porous media domain. Copyright © 2010 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Variability Models, Porous Media, Stencil

1. INTRODUCTION

Many real-world physical and chemical processes can be modeled using partial differential equations (PDEs). These processes arise in different areas, such as shallow-water equations [1], fluid dynamics [2], or porous-media applications [3]. To simulate these processes, grid-based methods and stencil-based solvers are commonly used.[†] In this article, we focus on the domain of porous-media applications, which offers a rich variety of models and of stencils to simulate the discretized equations describing the given applications. The overall goal of the article is to classify the models and stencils that are necessary to develop code generators and optimizers for real-world problems in this area.

Depending on the application scenario, the models have different characteristics considered in the simulation. For example, simple equations, such as the heat equation, assume a homogeneity in the porous media, while more complex equations, such as Darcy's law, consider heterogeneous computation domains. In general, equations of the porous-media domain may have many different properties, leading to models of different complexity, much like, the stencils that can be used to solve the equations appropriately. For applications, a stencil might work on scalar data and might

*Correspondence to: Alexander Grebhahn, Department of Informatics and Mathematics, University of Passau, Innstr. 33, 94032 Passau, Germany.
[†]There are other areas where problems on irregular domains are considered, such as certain fluid dynamics or structure mechanics applications. To model these irregular domains appropriately, unstructured grids are used, to which stencil techniques are not applicable.

have constant entries, while, for complex applications, it might be necessary to solve a matrix computation in each evaluation step.

In this article, we explore the variability of equations of the porous-media domain and the variability of the resulting stencils. Specifically, we highlight and characterize the most important differences among equations (e.g, whether a solution has to be transported through the domain or not) and among the stencils that are used to simulate them. Then, we use a variability-modeling technique to model the differences (and commonalities). To show that all of revealed differences among equations and stencils are relevant for the domain of porous-media equations, we investigate on four different equations: heat equation, Darcy's law, Advection-diffusion equation, and Richard's equation. For this set of equations, we present the influence of discretization techniques on the resulting stencils used to solve the respective equations. These examples expand to fairly complex real-world stencil problems, and we hope that they prove useful as a set of test models for compiler or domain specific language engineers. Additionally, we want aim at supporting application engineers in understanding the complexity and choices of stencil computations in the domain of porous media applications.

This article is structured as follows: First, we discuss the applicability of stencil-based methods and its relation to time-stepping schemes and linear solvers. We illustrate this with the well-known heat equation in Section 2. In Section 3, we discuss a range of different porous-media models and different methods for discretization. In Section 4, we introduce a variability-modeling technique and apply it to model the different choices of models and stencils in our domain of porous-media applications. In Section 5, we present a selection of stencils for different model variants of varying complexity. In Section 6, we discuss related work, and, with Section 7, we conclude this work.

2. THE HEAT EQUATION: STENCIL OPERATORS, MATRICES, AND LINEAR SOLVERS

Before discussing the domain of porous-media applications, let us briefly discuss the relation between models, discretization, matrices, linear solvers, and stencils. As most of the problems we discuss later are parabolic or elliptic equations, we start with the example of the heat equation.

The heat equation

$$\partial_t u(x,t) - \Delta u(x,t) = f(x,t) \qquad \text{on } \Omega \tag{1}$$

is the simplest parabolic equation and is closely related to the Poisson equation

$$-\Delta u(x) = f(x) \qquad \text{on } \Omega. \tag{2}$$

Both models are linear partial differential equations (PDEs) and share the same elliptic operator. There are close relations regarding the applicability of stencil methods. The heat equation (1) can be solved using either an explicit scheme or an implicit scheme. Both are based on the approximation of the derivative in time ∂_t by finite differences as

$$\partial_t u(x) \approx \frac{u(t+\tau) - u(t)}{\tau},$$

in the explicit case, the operator Δ is applied to u at the previous time t, yielding

$$u(t+\tau) = u(t) + \tau(\Delta u(x,t) + f(x,t)),$$

in the implicit case, it is applied at the current time $t+\tau$, that is,

$$u(t+\tau) = u(t) + \tau(\Delta u(x,t+\tau) + f(x,t+\tau)).$$

One way to solve the Poisson equation is to consider it as a stationary solution of the heat equation and to use an explicit Euler time-stepping scheme. This approach is slow but illustrates the close relation between linear solvers and explicit-time discretizations, as are very often considered when investigating stencil methods.

On a given grid, the PDEs are discretized and can then (at least, formally) be written in terms of a system matrix A and vectors. This leads either to a linear system ($Ax = b$) in the implicit case or to an explicit relation (e.g., in the `axpy` operation: $y = Ax + y$) between the matrix and the unknown solution vector x. In the case of a linear system, the matrix A needs to be inverted, which is done using a linear solver, such as the conjugate gradient (CG) method [4], ideally in combination with an efficient preconditioner, for example, a multigrid method [5]. To apply the CG method, it is only necessary to compute matrix-vector products; thus, the challenges of an efficient implementation is very similar for implicit and explicit time-discretization methods.

If the problem and the discretization exhibit enough structure, stencil methods can be applied to speed up the computation significantly, as compared to the classic approach, which actually assembles the matrix. The main reason for that is the low arithmetic intensity, since the matrix dominates the memory transfer costs, at least, when heterogeneous coefficients are involved. In the simplest case, we consider a 2D domain with a rectangular grid of size $N_x \times N_y$, using a finite-element discretization and ignore boundary conditions. In this case, when the unknowns are ordered lexicographically, the matrix A has a block structure with $N_y \times N_y$ blocks, where each of the blocks has a size of $N_x \times N_x$:

$$A = \frac{1}{3}\begin{pmatrix} T & O & & & \\ O & T & O & & \\ & O & \ddots & \ddots & \\ & & \ddots & & O \\ & & & O & T \end{pmatrix}$$

$$\text{where} \quad T = \begin{pmatrix} 8 & -1 & & & \\ -1 & 8 & -1 & & \\ & -1 & \ddots & \ddots & \\ & & \ddots & & -1 \\ & & & -1 & 8 \end{pmatrix}, \quad O = \begin{pmatrix} -1 & -1 & & & \\ -1 & -1 & -1 & & \\ & -1 & \ddots & \ddots & \\ & & \ddots & & -1 \\ & & & -1 & -1 \end{pmatrix}.$$

With each vertex in the grid, we can associate one particular matrix row. Each row, corresponding to an interior vertex, has the same structure describing the coupling to the neighbouring vertices:

$$\left[\ldots, -\frac{1}{3}, -\frac{1}{3}, -\frac{1}{3}, \ldots, -\frac{1}{3}, \frac{8}{3}, -\frac{1}{3}, \ldots, -\frac{1}{3}, -\frac{1}{3}, -\frac{1}{3}, \ldots\right].$$

This local coupling can be expressed as the well known 9-point stencil $\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$.
As said before, if the discretization of a PDE yields enough structure, stencil methods can be used. While the stencil for the heat equation using the finite-elements discretization looks simple, the stencil will change dramatically for more complex equations, such as Darcy's law. In Section 5, we will demonstrate how the stencils look like for different equations to solve them adequately and efficiently.

3. A FAMILY OF MODELS

Very often problems in porous media are described on structured grids – most often on uniform Cartesian grids, but, at least, on grids that are topologically structured. Such problems are very well-suited for stencil methods. What a particular stencil looks like, or whether stencil methods are applicable at all, depends on the considered physical phenomenon, the type of boundary conditions, and the particular numerical method we use. Let us give an overview of the variety of options available. We are aware that this list is not complete, but these are the options we consider the main options of our application domain.

3.1. Scope of the Application Domain

In real-world applications, often the phenomena that are of interest do not occur isolated, but as a combination of effects. Perhaps the most challenging problems nowadays are multi-phase-multi–component problems, where multiple fluid phases are modeled and, in these phases, multiple solutes are transported, which again can react with each other. In CO_2–water–oil systems, for example, the water and the oil are immiscible. The same applies for the two fluids and the super-critical CO_2. But, at the interface between water and CO_2, a small portion of the CO_2 can dilute in the water and be transported. Models considering thermal effects, such as changes in solutability or density due to temperature changes or coupling with thermo-mechanical effects, are of similar complexity. These types of problems and their models are subject to current research and beyond the scope here.

To solve these non-linear instationary problems it requires to further select a suitable time-stepping method and a non-linear solver. From the computational science point of view, the most critical component is the inner loop of the grid, this is where stencil approaches apply. All other components of the non-linear solver or the time-stepping method use coarse-grained interfaces and are thus not performance critical. We mention them here to emphasize that the application domain we consider cannot cover the whole scope of porous-media applications. Nevertheless, we have chosen a family of problems that we deem representative with respect to stencil computations.

3.2. Physical Phenomena

The models in porous-media applications range from simple elliptic equations to parabolic and nearly hyperbolic equations. Most models we describe in the following can be found in the classic textbooks of Jacob Bear [3, 6].

Depending on the complexity of the application, different strategies have to be used to solve it. For example, flow fields are described by *Darcy's law* and extensions of it (see (6)). Transport phenomena are in the simplest case diffusion or advection-diffusion systems of solutes. Often, multiple solutes are considered that eventually react with each other (e.g., nutrients, O_2 and bacteria). More complicated models consider multiple phases, such as water–gas or water–oil, and show a non-linear behavior, where the flow field changes over time due to the transport of the different phases.

Single-phase flow. Single-phase flows describe flow phenomena of a single fluid that fills the whole porous medium. The term 'single-phase flow' refers to the advective flow of this fluid. As we consider a conservative system (i.e., divergence-free velocity fields), this field can be represented as the gradient of the potential, which leads to Darcy's law

$$-\nabla \cdot [K(x)\nabla u(x)] = f(x) \qquad \text{on } \Omega, \tag{3}$$

which is basically the Poisson equation with heterogeneous coefficients. The particular challenge is that the coefficients can vary drastically and that the difference between the lowest and the highest coefficient can be up to 10^5 or 10^6. From the potential u, the flux can be computed as

$$\sigma(x) = K(x)\nabla u(x). \tag{4}$$

This relation can be used to reformulate the problem in mixed form as the system

$$\begin{aligned} -\nabla \cdot \sigma(x) &= f(x) && \text{on } \Omega, \\ K(x)\nabla u(x) &= \sigma(x) && \text{on } \Omega. \end{aligned}$$

Multi-phase flow. Multi-phase flows belong to the most important non-linear models in porous-media applications. They describe the flow of n immiscible fluids through the porous medium. Such fluids are typically oil and water, or oil, water, and gas. Each of the considered phases is governed by a mass-balance equation similar to the transport equation, where the transport velocity is described by Darcy's law. The PDE for a phase $\alpha \in [0, n)$ then reads:

$$\partial_t S_\alpha(x) - \nabla \cdot [K(S_\alpha(x))\nabla p_\alpha(x)] = 0 \qquad \text{on } \Omega, \tag{5}$$

where the saturation $S_\alpha \in [0, 1]$ describes the relative pore space occupied by phase α. Now, the permeability is no longer a constant, but depends on the phase saturation; also the permeability $K(S_\alpha)$ changes. Different physical models describe this relation (e.g., the Mualem law [7]). At the same time, constitutive laws describes the relation between phase pressures and phase saturations (e.g., the van-Genuchten model [8] or the Brooks–Corey model [9]). This allows us to eliminate one unknown from Equation (5).

For an individual phase, we can choose to solve either for its pressure or saturation. However, the pressure of a phase is not defined if the phase is absent; this has to be considered when solving the equation.

Note further that these constitutive relations are highly non-linear functions. Thus, the stencil now is no longer a classic stencil with fixed scalar values. Instead, to evaluate the stencil at a specific location, it is necessary to evaluate functions that define, for example, the saturation at the current location of the domain onto which the stencil is applied.

Solutes. Solutes in the fluid diffuse and get transported with the flow field. This is covered by the advection-diffusion equation

$$\partial_t c(x) - \nabla \cdot [D(x)\nabla c(x)] + \nabla \cdot [v(x)c(x)] = f(x) \qquad \text{on } \Omega. \tag{6}$$

There is a set of different approaches that can be used to solve this equation [10]. We discuss here one common approach using operator splitting, where the diffusion operator is solved implicitly and the advection operator explicitly: The advective term is hyperbolic. To be robust also in cases where the advection is the dominant process, it is necessary to apply stabilisation techniques, such as upwinding. These techniques take the local velocity field into account to guarantee a stable, physically-correct solution. For stencil codes, the down-side is that values and shape of the stencil change, depending on the velocity field.

If multiple solutes are considered, these can further interact with each other via chemical reactions. This leads to a system of advection-diffusion-reaction equations

$$\partial_t c(x) - \nabla \cdot [D(x)\nabla c(x)] + \nabla \cdot [v(x)c(x)] + R_i(c_0, \cdots) = f(x) \qquad \text{on } \Omega. \tag{7}$$

Again, these models can be solved using operator-splitting techniques. Depending on the reaction rates, these can be computed explicitly or by solving a non-linear system of equations.

Richard's equation. Richard's equation describes a simplified model of the two-phase flow for water and gas. When considering the flow of water in unsaturated soils, we can assume that the gas phase is always connected to the atmosphere, which implies that the gas-phase pressure is always atmospheric pressure. Consequently, we eliminate the pressure unknown of the gas phase and obtain a reduced problem.

As before, we can either eliminate pressure or saturation (or water content) using their constitutive relations. We present the formulation in terms of the water pressure p_w

$$C_w(p_w(x))\partial_t p_w(x) - \nabla \cdot [K_w(p_w(x))(\nabla p_w(x) - \rho g \mathbf{e}_z)] = 0 \qquad \text{on } \Omega, \tag{8}$$

with appropriate boundary conditions and a forcing term $\rho g \mathbf{e}_z$, due to gravity. The function $C_w(p_w) = \partial_{p_w} S_w(p_w)$ now describes the rate of change of saturation with respect to the pressure. We can also describe the permeability in terms of the pressure $K(p_w) = K(S_w(p_w))$. It is only a slight modification of the classical Darcy's law, but we consider it an important model problem, as it describes an instationary, non-linear system. Historically, Richard's equation builds upon previous work by Buckingham [11] on a non-linear dependence of the flux; together with mass conservation and the constitutive relative of p_w and S_w (e.g. the *Brooks-Corey*, or the *Van Genuchten-Mualem* model) one obtains (8). This formulation is also known as the non-conservative form. While there are other formulations of the same model, which are the preferable from the modelling point of view, we believe this to be an illustrative example regarding the applicability of stencil codes.

3.3. Boundary Conditions

So far, we have discussed different physical models. However, these models are only fully described when complemented with boundary information, initial conditions (in the case of instationary problems), and possible additional constraints.

Dirichlet Boundary Conditions. This type of boundary condition is essential for our type of models. In general, boundary conditions have to be specified to close the system. In some cases, additional constraints have be to added to the equation. Using Dirichlet boundary conditions, the solution is prescribed on the boundary. This implies that the unknowns associated with Dirichlet boundary conditions are fixed to a given value and, thus, are not unknown any more. As a result, different strategies can be used to handle unknowns. Either we keep them as entries in the solution vector, but make sure not to update them, or we remove them completely from the system, which implies that we have to alter the corresponding stencil of the interior unknowns. Here, we focus only on the second strategy.

If the boundary value is 0, *homogeneous Dirichlet boundary conditions* are derived. In this case, halo cells with a value of 0 can be applied at the boundary of the domain. If the boundary value is not equal to 0, *inhomogeneous Dirichlet boundary conditions* are considered.

Neumann Boundary Conditions. Neumann boundary conditions are often referred to as natural boundary conditions. They prescribe the flux (or numerical flux) over a boundary. In this sense, they are complementary to Dirichlet boundary conditions. For second-order PDEs, as we are considering here, it is not possible to prescribe flux and value at a given point on the boundary. For stationary problems, we observe a particular issue with Neumann boundary conditions.

As for Dirichlet conditions, we also know *homogeneous Neumann boundary conditions*. They have a flux of 0. Again, this leads to a case that can easily be handled using halo cells and mirroring the inside values. Again, if the flux is not 0, *inhomogeneous Neumann boundary conditions* are used.

In the case of pure Neumann boundary conditions, the solution of the PDE is not unique (i.e., it is only defined up to a constant). Thus, we need an additional constraint to close the system. The usual approach is to prescribe that the mean value of the solution has to be zero. Note that, for instationary problems, this is not an issue. Likewise, if one is using iterative Krylov solvers to solve the arising linear system [12], this is also not an issue, as the Krylov solver ensures that the mean value of the solution is not altered.

Periodic Boundary Conditions. Periodic boundary conditions are very common in the stencil community and easy to implement. These boundary conditions basically state that there is no physical boundary, but it is actually linked to the opposite side of the domain. In practice, they are not used too often. As with pure Neumann boundary conditions, we have the issue that the solution of stationary problems is only defined up to a constant.

Robin Boundary Conditions. Robin boundary conditions are a weighted combination of Dirichlet and Neumann boundary conditions. They describe a flux that depends on the current solution. Recalling the initial head equation, one can illustrate the Robin boundary conditions as a thin isolator. Outside of the domain, we assume a constant temperature, but the insulator reduces the heat exchange, such that we do not have the outside temperature at the boundary; instead, we have a heat flux proportional to the difference between interior and exterior temperature.

Outflow Boundary Conditions. The natural behavior at the outflow boundary for a transport equation is described by outflow boundary conditions. The transport equation is a first-order hyperbolic equation and, thus, the solution is only determined by the point that can be reached when following the streamlines backwards in time. In particular, this means that the boundary does not have any influence on the solution, if it is located downstream (i.e., the velocity is pointing out of the domain). Furthermore, there is no choice about where to impose this boundary condition, it has to be imposed exactly where the flow is going out of the domain.

3.4. Numerical Methods

To actually solve one of the models of Section 3.2 numerically, it is necessary to discretize them. The usual approach is to discretize independently in space and time. Different choices of the spatial discretization lead to different stencils, with sometimes very different properties.

Most of the existing methods can be found in standard text books in numerical simulations, such as by Alfio Quarteroni et al. [13] or Peter Knabner and Lutz Angerman [14]. In the case of more advanced methods, we will point to additional references.

Finite-Difference Methods. Finite-Difference methods are perhaps the oldest approach for discretizing PDEs. Given a Cartesian grid with a uniform spacing h in each direction, we seek for a solution at the vertices of the grid. First, the differential operators are approximated by some finite-difference representation or difference quotient (i.e., the right-sided approximation in 1D $\frac{\partial u(x_i)}{\partial x} \approx \frac{u_{i+1}-u_i}{h}$, or $\frac{\partial^2 u(x_i)}{\partial x^2} \approx \frac{u_{i+1}-2u_i+u_{i-1}}{h^2}$). This formulation of the application of a differential operator in terms a vertex value and its neighbours leads naturally to a stencil. It is possible to extend the method to higher-order approximations and to non-uniform grids, but this will, in general, lead to more complicated stencils. These methods are usually not used in porous-media applications, as they usually exhibit a large approximation error in the presence of large material jumps [15].

Finite-Volume Methods. Finite-Volume methods are closely related to finite-difference methods, but do not start from the PDE, but from the corresponding integral formulation. This formulation is obtained by applying the Gauss theorem for a given control volume (e.g., the cell of the grid). Instead of the relation between the solution in a point and the derivatives, one obtains a relation between the integral of the solution over the control volume and fluxes through the boundary. In the case of the heat equation, these fluxes take the form $\nabla u \cdot n$, with n denoting the outwards pointing unit vector. To approximate ∇u, a finite-difference approximation is employed. Such flux-based formulations are particularly useful for advection-dominated problems, where the diffusion has only minimal impact and thus the PDE is nearly hyperbolic.

Finite-Element Methods. Finite-element methods are widely used in scientific and engineering applications. A detailed introduction into finite-element methods, especially continuous finite-element methods and mixed finite-element methods is available elsewhere [16]. Instead of the partial differential equation itself, they start with its weak form, which allows for a rigorous analysis of the equations and their solution. For instance, take Darcy's law (3): Multiplying with a test function v and integrating over the whole domain Ω yields

$$-\int_\Omega \nabla \cdot [K(x)\nabla u(x)]v(x)dx = \int f(x)v(x)dx\,.$$

Applying the divergence theorem gives

$$\underbrace{\int_\Omega [K(x)\nabla u(x)] \cdot \nabla v(x)dx}_{=:a(u,v)} = \underbrace{\int f(x)v(x)dx}_{=:\ell(v)}\,, \tag{9}$$

which defines the bilinear form a and a linear functional ℓ.

Continuous Galerkin. For the numerical solution, Equation (9) is discretized by dividing Ω into finite subdomains, the finite-elements. Often the domain is approximated by triangles or by rectangles. In continuous Galerkin methods, the function is assumed to be continuous, imposing conditions on the values at the element boundaries. The bilinear form a and the linear form ℓ are then approximately evaluated on the elements by quadrature. Different types of elements are used to approximate various derivatives or to use different orders of approximation accuracy.

Mixed Finite-Element Methods. Mixed-finite elements originate from the discretization of sattlepoint problems, as they arise when PDEs with constraints are to be solved. They naturally allow for the use of different types of elements for the different variable types that are part of the system of PDEs to be solved. This corresponds to the different function spaces. In porous-media applications, mixed-finite elements can be used to solve systems of PDEs. For example, Darcy's law (3) can be reformulated as

$$\begin{aligned} -\nabla\sigma(x) &= f(x) \qquad \text{on } \Omega, \\ K(x)\nabla u(x) &= \sigma(x) \qquad \text{on } \Omega. \end{aligned}$$

The artificial variable σ, which was introduced to Equation (4) as a system in a mixed-finite element setting, can be discretized using other finite-elements than those used for the discretization of u.

Discontinuous Galerkin Methods. A special kind of finite-element methods are discontinuous Galerkin methods. They are as flexible as the finite-element method, but, similar to the finite-volume methods, they strongly relate to the integral formulation of the PDE. Again, we seek for the solution that approximates the exact solution in the best way, in the sense, that the error is orthogonal to our discrete function space; we use a Galerkin formulation. The big difference to discontinuous Galerkin methods is that we do not use continuous functions to approximate the solution, but we take functions that are continuous on each element, but might be discontinuous between grid elements. We derive a weak formulation by testing with arbitrary functions of the same type, integration by parts on each element leads to a flux based formulation. As this formulation is in many cases non-symmetric and not stable, additional terms are added, which vanish in the exact solution, but lead to a stable method. A very common discontinuous Galerkin discretization is the Symmetric Interior Penalty Galerkin (SIPG) method, which goes back to a paper by Wheeler [17]. A general overview over discontinuous Galerkin methods can be found elsewhere [18].

3.5. Summary

Overall, many different ways exist to model the problem in porous-media applications, depending on the physical phenomena that shall be captured. Furthermore, there are many approaches to discretize the underlying model to make it accessible numerically. This results in a large variety of different stencil codes for these applications with different properties that have to be captured by a variability model.

4. VARIABILITY MODELING OF POROUS-MEDIA APPLICATIONS AND THEIR STENCILS

After presenting different models of the domain of porous-media applications together with their properties, we now use a variability modeling technique to provide application engineers an overview of decisions they have to make and properties they have to consider when selecting a suitable stencil for a given mathematical problem.

For the purpose of variability modeling, we use *feature models* and, as their graphical representation, *feature diagrams* [19, 20]. Each feature of a feature model describes a specific domain abstraction[‡], which can be a commonality or a difference between models of the domain,

[‡]Here, domain is referred to the application domain not the computation domain.

for example, different boundary conditions are needed to model properties of the application. To separate cause and effect when selecting a stencil for a given problem, we model the corresponding variability in two separate models: one model describing the variability based on the differences in the applications and one model describing the variability in the implementation of the stencils to solve the application. In Figure 1, we present the feature diagram for the application properties and, in Figure 2, for the variability of the stencils. However, we consider only common cases in the feature diagrams to represent for a common set of application scenarios, and we ignore special cases because they would make the variability model very complex without any benefits for application engineers.

To define relationships between features, feature diagrams have a tree-like structure inducing a parent-child relationship between features. In general, a feature is either *mandatory* or *optional*. While mandatory features have to be selected, if their parent feature is selected, optional features can either be selected or deselected, if their parent feature is selected. An example of an optional feature is feature *Solute Transport* in Figure 1. Depending on the mathematical problem, it is necessary to consider the transport of a solute or not. In contrast, it is always necessary to apply a discretization on the PDE. Thus, *discretization* is a mandatory feature.

Features can be grouped, for instance, in alternatives to define that exactly one feature of the *alternative group* has to be selected. The other kind of groups are *or groups*, in which, at least, one feature of the group has to be selected, but multiple are possible. Examples for the different kinds of groups can be seen in Figure 1: Depending on the problem to solve, it might be necessary to select different boundary conditions at different parts of the boundary of the domain. As a result, multiple boundary conditions can be selected at the same time, but, at least, one has to be selected. In contrast, it is necessary to use exactly one discretization method. Thus, the *boundary condition* is in an *or group*, while the discretization method is in an *alternative group*.

In addition to the feature diagram, one can define arbitrary propositional formulas over features to constraint variability further. The two most prominent constraints are *imply* and *exclude*. If one feature implies another, the second one has to be selected if the first one is selected. If one feature excludes another, they cannot be selected at the same time. For example, if the *Robin* boundary condition is used in an application, *Special Stencils* have to be used at the boundaries of the domain (Figure 2). As in our scenario properties of the application have an influence on the selection of a suitable stencil, we have to define constraints between the two models. For these inter-model constraints, we use a prefix notation to disambiguate between two features with the same name. For example, we write *Dirichlet.Homogeneous* to refer to a homogeneous Dirichlet boundary condition and *Neumann.Homogeneous* to refer to a homogeneous Neumann boundary condition.

4.1. Variability of the Mathematical Model

To numerically solve a PDE, one suitable discretization technique has to be selected. Here, the characteristics of the equation have to be considered, for example, whether it is a hyperbolic, elliptic, or parabolic. In Figure 1, we give the feature model and in Table I, we provide more explanation.

The discretization technique has to be applied to the whole computation domain of the application, leading to positive system matrices. To fulfill the properties at the boundary of the domain, it is necessary to select specific boundary conditions. Here, it might be necessary to select different conditions on disjoint parts of the boundary, which is known as mixed-boundary conditions. Overall, each point of the entire boundary has to be covered by exactly one boundary condition.

If multiple phases are considered, it is important to identify whether the phases interact with each other or not. In real-world applications, it might happen that some of the phases interact with each other, while the flow of other phases can be computed independently. However, to keep the model simple, we do not include this property (see Section 3.2).

Furthermore, the application might define a solute transport, which has to be considered in the discretization. There, it might be possible that not only one solute is transported through the domain but multiple solutes have to be considered.

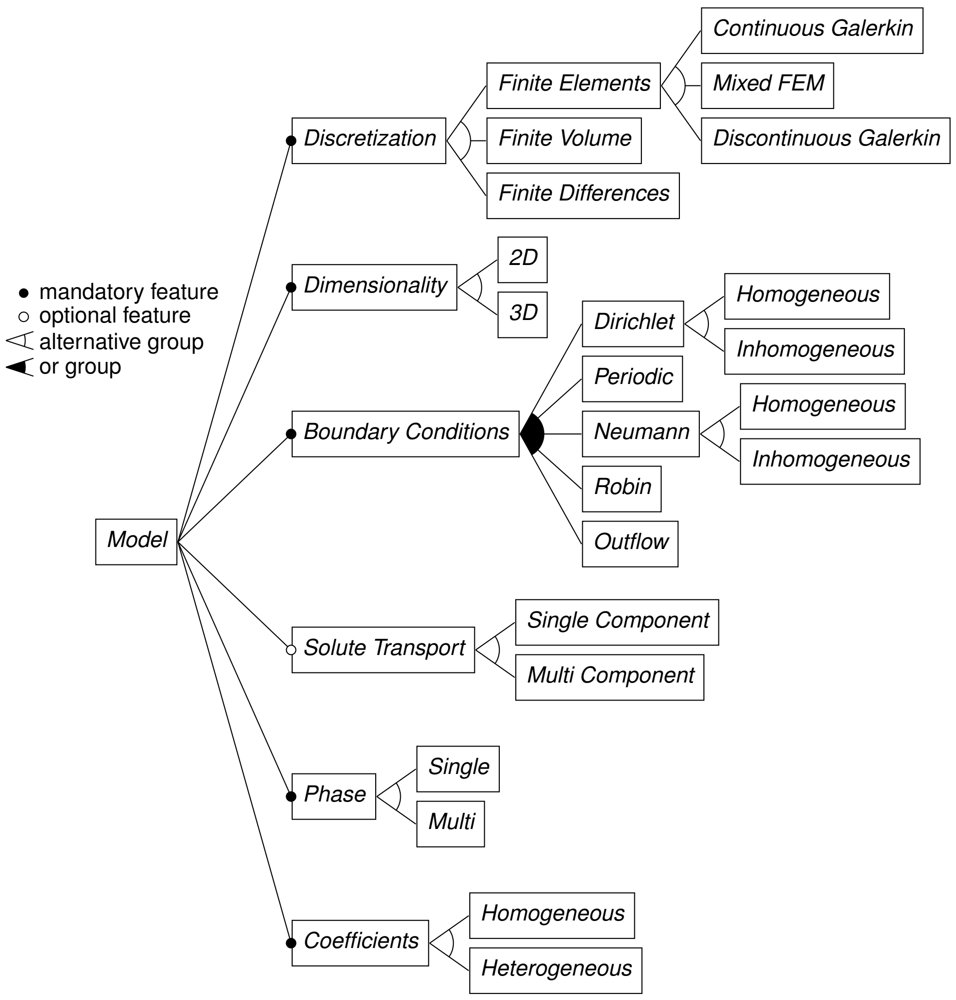

Figure 1. Feature diagram of the mathematical part of the porous media domain.

It is also necessary to consider whether the equation has homogeneous or heterogeneous coefficients. In the first case, the stencil will stay constant throughout the domain; in the second case, the implementation has to support heterogeneous coefficients. This heterogeneity can be supported in two different ways. First, the coefficients can be stored as a separate field, which leads to a large but constant stencil. Second, when considering the coefficients as an arbitrary scalar function, the stencil cannot be written with constant values anymore, but each entry is itself a function, processing the values of neighbouring cells and the coefficients' functions.

To conclude, all of these features have a considerable influence on the structure of the stencils that can be used to solve the equation.

4.2. *Variability of Stencils*

In stencil applications, such as multigrid methods, it is desired to have the code run as fast as possible. To achieve this goal, it is possible to use optimization strategies to parallelize a stencil computation using automatic code transformation [21, 22], or to use techniques to improve the cache reuse of a stencil computation, such as temporal or spacial blocking [23, 24, 25]. However, all of these techniques have to preserve dependencies between elements of the computation domain not to affect the result of the computation. Thus, optimization techniques typically do not modify the shape of a stencil or the amount of data the stencil is working on, which is defined by the

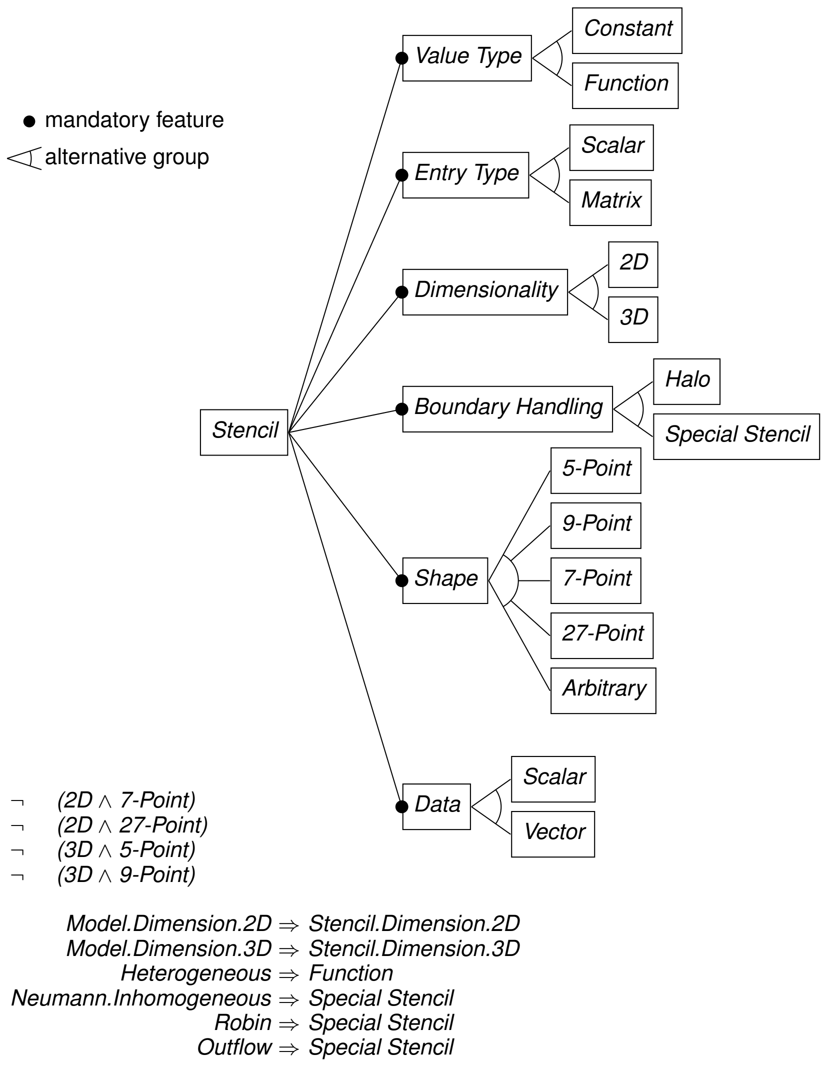

Figure 2. Feature diagram of stencils of the domain of porous-media applications.

application on the mathematical level. Here, we do not focus on optimization strategies, but on the general structure of a stencil to solve a given application scenario.

In Figure 2, we give the feature model of the properties of a stencil and, in Table II, we provide additional explanations. Depending on the characteristics of the application, the data on which the stencil is working on might look completely differently. For example, for equations such as the heat equation, the stencil has constant entries, while for more complex equation, it is necessary to evaluate complex functions, to compute one entry of the stencil.

Based on the dimensionality of the problem, the dimensionality of the stencil is predefined. That is, if the equation is defined for a three-dimensional space, the shape of the stencils have to be defined for this dimensionality, as well. For the stencil shapes presented in Figure 2, this means that, for example, a 7-point stencil as well as a 27-point stencil cannot be applied to a problem defined for a two-dimensional domain. Beside the dimensionality, the shape of the stencil also depends on the discretization technique used. For example, as we will explain in Section 5 in more detail, if a finite-differences discretization is used on the heat equation in a two-dimensional domain, a 5-point stencils has to be used, while a 9-point stencil is needed, if the finite-element discretization is used instead. Additional, specific stencils are needed, if the boundary conditions are not handled by prescribing the appropriate values in the solution vector.

Also, depending on the application, the values of a stencil will look completely differently. As already stated in Section 4.1, if heterogeneous coefficients are considered and the coefficients are not stored in a separate field, the stencil does not have constant entries. Instead, it is necessary to evaluate a function at every step for each of the entries. We refer to this characteristic as *Value Type* in Figure 2. If multiple solutes are considered in the application that do not interact with each other, the stencil might consider both of them. Thus, the stencil has to work on vector data instead of scalar data.

Finally, depending on the number of phases (single phase vs. multi phase) and on whether the phases interact with each other, the stencil has to work on *Scalar* or on *Matrix* entries. The size of the matrix is defined by the number of solutes that have to be considered at the same time. Scalar entries can be used if the stencil only works for one phase, while matrix entries are needed if multiple interacting phases are considered in one stencil.

5. STENCILS FOR POROUS-MEDIA APPLICATIONS

In this section, we present a set of PDEs of varying complexity for different porous-media applications. For each of the applications, we discuss which of the features of Section 4 is relevant to solve it. To this end, we discuss how the characteristics of the applications influences structure of the stencils. Furthermore, we state constraints between features at the mathematical level (see Section 4.1) and features of the stencil (see Section 4.2). Based on these constraints, we give a presence condition for each of the feature models and for each of the equations: These presence conditions describe–via logical formulas–which of features has to be selected to solve a equation adequately.

For illustration, let us have a brief look at the heat equation: In this equation, only one phase is considered and no solute transport has to be simulated. Furthermore, the computation domain only has homogeneous coefficients. As result, the two features *Homogeneous* and *Single* are selected for the equation, and the feature *Solute Transport* has to be deselected, which is marked with a logical `not` ($\neg$) in the presence condition. So, for these three features, the presence condition is: *Single* $\wedge$ *Homogeneous* $\wedge \neg$ *Solute Transport*

In the following, we assume that we want to solve problems on a two-dimensional structured grid with constant grid spacing h in each direction. The values of the concentration c are sought for on the grid points of this grid. To compute the solution, the different operators originating from discretizing the models of Section 3.2 by the techniques of Section 3.4 have to be applied to the field where the approximate values of c are stored. This has to be done to obtain the value in the next time step or to obtain a better approximation when a system has to be solved in an implicit scheme.

Heat equation. The simplest example for an equation with constant coefficients is the heat equation, see Equation (1). Its implementation involves the discretization of the Laplace operator, to evaluate it during the explicit time-stepping scheme or an iterative solver in the case that an implicit scheme is chosen.

i) The simplest discretization is the finite-difference discretization, leading to the 5-point stencil

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}. \tag{10}$$

If the boundary conditions are not handled by prescribing the appropriate values in the solution vector, special stencils have to be applied at the boundaries. For example, when Dirichlet boundary conditions are imposed, the stencil in the top left corner of the domain is given by

$$\frac{1}{h^2}\begin{bmatrix} & 4 & -1 \\ & -1 & \end{bmatrix}; \tag{11}$$

similarly, for Neumann boundary conditions, we obtain the following stencil at the right boundary of the domain:

$$\frac{1}{h^2}\begin{bmatrix} & -1 \\ -1 & 3 \\ & -1 \end{bmatrix}. \tag{12}$$

Note that the location of the center element is now only unique when the location inside of the domain is known and the convention is used that the stencil is shifted to the boundary. So, for the finite-differences discretization on the heat equation, the following presence conditions for the two feature models can be given:

Model: *Finite-Differences* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single* $\wedge$ *Homogeneous* $\wedge$ $\neg$ *Solute Transport*

Stencil: *Constant* $\wedge$ *Entry-Type.Scalar* $\wedge$ *2D* $\wedge$ (*Special-Stencil* $\vee$ (*Dirichlet* $\Rightarrow$ (*Halo* $\vee$ *Special-Stencil*))) $\wedge$ *5-Point* $\wedge$ *Data.Scalar*

ii) The more advanced finite-element discretization leads to the following 9-point stencil:

$$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}. \tag{13}$$

As in the finite-difference case, the stencil has to be altered to accommodate boundary conditions. This leads to similar modifications as we have presented for the finite-difference discretization.

In all stencils, it is possible to work on scalar data because the heat equation considers only one phase, and it is not necessary to consider functions as entries of the stencil. Additionally, no solute transport has to be considered. As a result, if we use a finite-element discretization, the following presence conditions describe the variability that have to be considered for the heat equation:

Model: *Finite-Elements* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single* $\wedge$ *Homogeneous* $\wedge$ $\neg$ *Solute Transport*

Stencil: *Constant* $\wedge$ *Entry-Type.Scalar* $\wedge$ *2D* $\wedge$ (*Special-Stencil* $\vee$ (*Dirichlet* $\Rightarrow$ (*Halo* $\vee$ *Special-Stencil*))) $\wedge$ *9-Point* $\wedge$ *Data.Scalar*

Darcy's law. In contrast to the heat equation, heterogeneous coefficients have to be considered in Darcy's law, which are described using the value of K at positions shifted by half of the volume's side length relative to the location of the stencil.

i) One natural way to discretize the Laplace operator in Equation (3) is using finite volumes, leading to the stencil

$$\begin{bmatrix} & s_n & \\ s_w & s_c & s_e \\ & s_s & \end{bmatrix},$$

where

$$\begin{aligned} s_n &= -K(x, y + h/2), \\ s_w &= -K(x - h/2, y), \\ s_e &= -K(x + h/2, y), \\ s_s &= -K(x, y - h/2), \\ s_c &= -(s_n + s_w + s_e + s_s). \end{aligned}$$

Using the position of K, it is no longer possible to use scalar values in the stencil. Instead, the parameters are given as function evaluations of K at the appropriate positions. The center of the current cell is denoted by (x, y). The factor $1/h^2$, which is missing in comparison to Equation (10), is moved to the right hand side, which is given as $h^2 f$. This is a result of the integration of the right-hand side over a finite volume, which, in our case, is a square with side length h.

Model: *Finite-Volumes* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single* $\wedge$ *Heterogeneous* $\wedge$ $\neg$ *Solute Transport*

Stencil: *Function* $\wedge$ *Entry-Type.Scalar* $\wedge$ *2D* $\wedge$ (*Special-Stencil* $\vee$ (*Dirichlet* $\Rightarrow$ (*Halo* $\vee$ *Special-Stencil*))) $\wedge$ *5-Point* $\wedge$ *Data.Scalar*

ii) In other applications, K is not given as a function, but rather as a field of its values. In this case, we obtain

$$\begin{bmatrix} & -K_{i,j+1} & \\ -K_{i-1,j} & K_{i,j+1} + K_{i-1,j} + K_{i+1,j} + K_{i,j-1} & -K_{i+1,j} \\ & -K_{i,j-1} & \end{bmatrix}.$$

Here, we assume that the field of the values of K is *staggered*, that is, the nodes of the field containing K are located at the cell centers of the field used to store u.

Again, on the boundary, modifications are necessary, depending on the type of boundary condition and on whether the values are prescribed in the solution vector. In the latter case, this results in missing stencil entries like the stencils for the corner (11) and the right boundary (12) in the heat equation case, as considered before. This leads us to the following presence conditions:

Model: *Finite-Volumes* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single* $\wedge$ *Heterogeneous* $\wedge$ $\neg$ *Solute Transport*

Stencil: *Constant* $\wedge$ *Entry-Type.Scalar* $\wedge$ *2D* $\wedge$ (*Special-Stencil* $\vee$ (*Dirichlet* $\Rightarrow$ (*Halo* $\vee$ *Special-Stencil*))) $\wedge$ *5-Point* $\wedge$ *Data.Scalar*

iii) Another option of discretizing the Laplace operator with non-constant coefficients is the use of finite-elements as before, yielding

$$\frac{1}{h^2}\begin{bmatrix} s_{nw} & s_n & s_{ne} \\ s_w & s_c & s_e \\ s_{sw} & s_s & s_{se} \end{bmatrix},$$

where

$$\begin{aligned}
s_{nw} &= -\frac{1}{3}K(x-h/2, y+h/2)\\
s_n &= -\frac{1}{6}\left(K(x-h/2, y+h/2) + K(x+h/2, y+h/2)\right)\\
s_{ne} &= -\frac{1}{3}K(x+h/2, y+h/2)\\
s_w &= -\frac{1}{6}\left(K(x-h/2, y+h/2) + K(x-h/2, y-h/2)\right)\\
s_e &= -\frac{1}{6}\left(K(x+h/2, y+h/2) + K(x+h/2, y-h/2)\right)\\
s_{sw} &= -\frac{1}{3}K(x-h/2, y-h/2)\\
s_s &= -\frac{1}{6}\left(K(x-h/2, y-h/2) + K(x+h/2, y-h/2)\right)\\
s_{se} &= -\frac{1}{3}K(x+h/2, y-h/2)\\
s_c &= -(s_{nw} + s_n + s_{ne} + s_w + s_e + s_{sw} + s_s + s_{se}).
\end{aligned}$$

Due to heterogeneity in the coefficients, this stencil again depends on the value of function K at positions close to the current vertex position (x, y). In general, an implementation with a field K is possible like in the finite-volume case. Note that, in the case of constant coefficients, we obtain the stencil given in Equation (13). This results in the following presence conditions:

Model: *Finite-Elements* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single* $\wedge$ *Heterogeneous* $\wedge$ $\neg$ *Solute Transport*

Stencil: *Function* $\wedge$ *Entry-Type.Scalar* $\wedge$ *2D* $\wedge$ (*Special-Stencil* $\vee$ (*Dirichlet* $\Rightarrow$ (*Halo* $\vee$ *Special-Stencil*))) $\wedge$ *9-Point* $\wedge$ *Data.Scalar*

iv) In many applications, K is varying strongly and with high contrast. This poses numerical difficulties, for example, for vertex-centered finite volumes or finite elements. One option to overcome this problem is the use of discontinuous Galerkin finite elements [26, 27]. In contrast to the stencils considered so far, discontinuous Galerkin methods introduce several degrees of freedom for the same entity, in this case, a grid cell. From the technical structure, the situation is comparable to a reaction diffusion system with many components. The solution is described by cell-local polynomials, and the unknowns are the coefficients of the polynomial basis. Continuity of the solution is not enforced strongly, but only using penalty terms. Considering a structured grid, it is still possible to handle such discretizations using stencils. Instead of scalar stencils, one now obtains block stencils with small matrices in each entry.

The general structure is the same as for a finite-volume discretization: a 5-point stencil. The individual entries are small dense matrices and their size depends on the polynomial order of the approximation

$$\begin{bmatrix} & -S_n & \\ -S_w & V+S & -S_e \\ & -S_s & \end{bmatrix}. \tag{14}$$

The stencil contains a set of terms S_* evaluated on the set of faces, the skeleton, and one term evaluated in the cell, considering the volume contributions V. The skeleton terms contain contributions arising from the model and the penalty terms J, which is necessary to obtain a stable method.

We consider a first-order weighted symmetric interior penalty Galerkin discretization as introduced by Ern et al. [28]. Using bilinear shape functions, we have a choice regarding the actual basis and the monomials $\{1, x, y, xy\}$, which yields 4×4 matrices. For this choice, the volume contribution reads

$$V = \frac{6}{K_c}\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & 3 \\ 0 & 0 & 6 & 3 \\ 0 & 3 & 3 & 4 \end{pmatrix}.$$

The skeleton contribution to the diagonal entry consists of the consistency and symmetry terms C, the penalty term J, and the term

$$S = C + J,$$

with

$$C = \frac{K_c}{6}\begin{pmatrix}
0 & \frac{6K_c}{K_c+K_w} - \frac{6K_c}{K_c+K_e} & \frac{6K_c}{K_c+K_s} - \frac{6K_c}{K_c+K_n} & \begin{matrix}\frac{3K_c}{K_c+K_s} - \frac{3K_c}{K_c+K_n} \\ + \frac{3K_c}{K_c+K_w} - \frac{3K_c}{K_c+K_e}\end{matrix} \\
\frac{6K_c}{K_c+K_w} - \frac{6K_c}{K_c+K_e} & -\frac{12K_c}{K_c+K_e} & \begin{matrix}\frac{3K_c}{K_c+K_s} - \frac{3K_c}{K_c+K_n} \\ + \frac{3K_c}{K_c+K_w} - \frac{3K_c}{K_c+K_e}\end{matrix} & \frac{2K_c}{K_c+K_s} - \frac{2K_c}{K_c+K_n} - \frac{6K_c}{K_c+K_e} \\
\frac{6K_c}{K_c+K_s} - \frac{6K_c}{K_c+K_n} & \begin{matrix}\frac{3K_c}{K_c+K_s} - \frac{3K_c}{K_c+K_n} \\ + \frac{3K_c}{K_c+K_w} - \frac{3K_c}{K_c+K_e}\end{matrix} & -\frac{12K_c}{K_c+K_n} & -\frac{6K_c}{K_c+K_n} + \frac{2K_c}{K_c+K_w} - \frac{2K_c}{K_c+K_e} \\
\begin{matrix}\frac{3K_c}{K_c+K_s} - \frac{3K_c}{K_c+K_n} \\ + \frac{3K_c}{K_c+K_w} - \frac{3K_c}{K_c+K_e}\end{matrix} & \frac{2K_c}{K_c+K_s} - \frac{2K_c}{K_c+K_n} - \frac{6K_c}{K_c+K_e} & -\frac{6K_c}{K_c+K_n} + \frac{2K_c}{K_c+K_w} - \frac{2K_c}{K_c+K_e} & -\frac{4K_c}{K_c+K_n} - \frac{4K_c}{K_c+K_e}
\end{pmatrix},$$

and

$$J = \frac{\eta K_c}{6}\begin{pmatrix}
\begin{matrix}\frac{12K_s}{K_c+K_s} + \frac{12K_n}{K_c+K_n} \\ + \frac{12K_w}{K_c+K_w} + \frac{12K_e}{K_c+K_e}\end{matrix} & \frac{6K_s}{K_c+K_s} + \frac{6K_n}{K_c+K_n} + \frac{12K_e}{K_c+K_e} & \frac{12K_n}{K_c+K_n} + \frac{6K_w}{K_c+K_w} + \frac{6K_e}{K_c+K_e} & \frac{6K_n}{K_c+K_n} + \frac{6K_e}{K_c+K_e} \\
\frac{6K_s}{K_c+K_s} + \frac{6K_n}{K_c+K_n} + \frac{12K_e}{K_c+K_e} & \frac{4K_s}{K_c+K_s} + \frac{4K_n}{K_c+K_n} + \frac{12K_e}{K_c+K_e} & \frac{6K_n}{K_c+K_n} + \frac{6K_e}{K_c+K_e} & \frac{4K_n}{K_c+K_n} + \frac{6K_e}{K_c+K_e} \\
\frac{12K_n}{K_c+K_n} + \frac{6K_w}{K_c+K_w} + \frac{6K_e}{K_c+K_e} & \frac{6K_n}{K_c+K_n} + \frac{6K_e}{K_c+K_e} & \frac{12K_n}{K_c+K_n} + \frac{4K_w}{K_c+K_w} + \frac{4K_e}{K_c+K_e} & \frac{6K_n}{K_c+K_n} + \frac{4K_e}{K_c+K_e} \\
\frac{6K_n}{K_c+K_n} + \frac{6K_e}{K_c+K_e} & \frac{4K_n}{K_c+K_n} + \frac{6K_e}{K_c+K_e} & \frac{6K_n}{K_c+K_n} + \frac{4K_e}{K_c+K_e} & \frac{4K_n}{K_c+K_n} + \frac{4K_e}{K_c+K_e}
\end{pmatrix}.$$

The off-diagonals again comprise consistency and penalty terms, but for simplicity, we just list the sum for each neighbouring cell:

$$S_s = \frac{K_c}{6}\begin{pmatrix}
\frac{12\eta K_s}{K_c+K_s} & \frac{6\eta K_s}{K_c+K_s} & \frac{12\eta K_s-6K_c}{K_c+K_s} & \frac{6\eta K_s-3K_c}{K_c+K_s} \\
\frac{6\eta K_s}{K_c+K_s} & \frac{4\eta K_s}{K_c+K_s} & \frac{6\eta K_s-3K_c}{K_c+K_s} & \frac{4\eta K_s-2K_c}{K_c+K_s} \\
\frac{6K_c}{K_c+K_s} & \frac{3K_c}{K_c+K_s} & \frac{6K_c}{K_c+K_s} & \frac{3K_c}{K_c+K_s} \\
\frac{3K_c}{K_c+K_s} & \frac{2K_c}{K_c+K_s} & \frac{3K_c}{K_c+K_s} & \frac{2K_c}{K_c+K_s}
\end{pmatrix}, \qquad
S_n = \frac{K_c}{6}\begin{pmatrix}
\frac{12\eta K_n}{K_c+K_n} & \frac{6\eta K_n}{K_c+K_n} & \frac{6K_c}{K_c+K_n} & \frac{3K_c}{K_c+K_n} \\
\frac{6\eta K_n}{K_c+K_n} & \frac{4\eta K_n}{K_c+K_n} & \frac{3K_c}{K_c+K_n} & \frac{2K_c}{K_c+K_n} \\
\frac{12\eta K_n-6K_c}{K_c+K_n} & \frac{6\eta K_n-3K_c}{K_c+K_n} & \frac{6K_c}{K_c+K_n} & \frac{3K_c}{K_c+K_n} \\
\frac{6\eta K_n-3K_c}{K_c+K_n} & \frac{4\eta K_n-2K_c}{K_c+K_n} & \frac{3K_c}{K_c+K_n} & \frac{2K_c}{K_c+K_n}
\end{pmatrix},$$

$$S_w = \frac{K_c}{6}\begin{pmatrix}
\frac{12\eta K_w}{K_c+K_w} & \frac{12\eta K_w-6K_c}{K_c+K_w} & \frac{6\eta K_w}{K_c+K_w} & \frac{6\eta K_w-3K_c}{K_c+K_w} \\
\frac{6K_c}{K_c+K_w} & \frac{6K_c}{K_c+K_w} & \frac{3K_c}{K_c+K_w} & \frac{3K_c}{K_c+K_w} \\
\frac{6\eta K_w}{K_c+K_w} & \frac{6\eta K_w-3K_c}{K_c+K_w} & \frac{4\eta K_w}{K_c+K_w} & \frac{4\eta K_w-2K_c}{K_c+K_w} \\
\frac{3K_c}{K_c+K_w} & \frac{3K_c}{K_c+K_w} & \frac{2K_c}{K_c+K_w} & \frac{2K_c}{K_c+K_w}
\end{pmatrix}, \qquad
S_e = \frac{K_c}{6}\begin{pmatrix}
\frac{12\eta K_e}{K_c+K_e} & \frac{6K_c}{K_c+K_e} & \frac{6\eta K_e}{K_c+K_e} & \frac{3K_c}{K_c+K_e} \\
\frac{12\eta K_e-6K_c}{K_c+K_e} & \frac{6K_c}{K_c+K_e} & \frac{6\eta K_e-3K_c}{K_c+K_e} & \frac{3K_c}{K_c+K_e} \\
\frac{6\eta K_e}{K_c+K_e} & \frac{3K_c}{K_c+K_e} & \frac{4\eta K_e}{K_c+K_e} & \frac{2K_c}{K_c+K_e} \\
\frac{6\eta K_e-3K_c}{K_c+K_e} & \frac{3K_c}{K_c+K_e} & \frac{4\eta K_e-2K_c}{K_c+K_e} & \frac{2K_c}{K_c+K_e}
\end{pmatrix}.$$

For this equation, we give the following presence conditions:

Model: *Discontinuous Galerkin* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single* $\wedge$ *Heterogeneous* $\wedge$ $\neg$ *Solute Transport*

Stencil: *Function* $\wedge$ *Entry-Type.Matrix* $\wedge$ *2D* $\wedge$ *Special-Stencil* $\wedge$ *5-Point* $\wedge$ *Data.Scalar*

Advection-diffusion with operator splitting. So far, we did not take into account the advective term $\nabla \cdot [v(x)c(x)]$ of Equation (6). To do so, Equation (6) is split into a purely diffusive and a purely advective part, yielding

$$\begin{aligned}\partial_t c(x) &= \underbrace{\nabla \cdot [D(x)\nabla c(x)] + f(x)}_{=:\mathcal{L}_{\text{diffusion}}} \underbrace{-\nabla \cdot [v(x)c(x)]}_{=:\mathcal{L}_{\text{advection}}} \\ &= \mathcal{L}_{\text{diffusion}} + \mathcal{L}_{\text{advection}}.\end{aligned}$$

This technique is called operator splitting. It has been introduced to be able to use different time integration schemes that are suitable for the respective parts.

i) To discretize $\mathcal{L}_{\text{diffusion}}$, the techniques described for Darcy's law can be used. When finite-differences are used to discretize $\mathcal{L}_{\text{diffusion}}$, the operator $\mathcal{L}_{\text{advection}}$ can be discretized using an upwind scheme, where it is approximated by

$$\begin{aligned}\mathcal{L}_{\text{advection}} &= -\nabla \cdot [v(x,y)c(x,y)] \\ &\approx \frac{1}{h}\left(v_x(x,y)c(x,y) - (v_x(x,y)^+ c(x-h,y) + v_x(x,y)^- c(x+h,y))\right) \\ &\qquad + \frac{1}{h}\left(v_y(x,y)c(x,y) - (v_y(x,y)^+ c(x,y-h) + v_y(x,y)^- c(x,y+h))\right),\end{aligned}$$

where $v_*(x,y)^+ = \max(v_*(x,y),0)$, $v_*(x,y)^- = \min(v_*(x,y),0)$.

This yields the stencil

$$\frac{1}{h}\begin{bmatrix} & -v_y(x,y)^- & \\ -v_x(x,y)^+ & v_x(x,y)+v_y(x,y) & -v_x(x,y)^- \\ & -v_y(x,y)^+ & \end{bmatrix}.$$

The time evolution is now computed by advancing the current concentration using either diffusion or using the opposite advection only, obtaining an intermediate result, which is afterwards advanced using advection or diffusion only. In both cases, the same time step is used, so the intermediate result is not the result at a half time step.

For this equation, we give the following presence conditions:

Model: *Finite-Differences* $\wedge$ *2D* $\wedge$ (*Dirichlet* $\vee$ *Neumann*) $\wedge$ *Single Component* $\wedge$ *Single* $\wedge$ *Heterogeneous*

Stencil: *Function* $\wedge$ *Scalar* $\wedge$ *2D* $\wedge$ *Special-Stencil* $\wedge$ *5-Point* $\wedge$ *Data.Scalar*

Richard's equation. For Richard's Equation (8), we assume the simplified setup of homogeneous material parameters, where K depends on the value of p_w at the current location, not only on the location itself.

i) This results in a non-linearity that, when being discretized using finite volumes, yields the stencil

$$\begin{bmatrix} & s_n & \\ s_w & s_c & s_e \\ & s_s & \end{bmatrix},$$

where

$$\begin{aligned}s_i &= -K(p^{\uparrow}_{wi}) \qquad \forall i \in \{n,w,e,s\}, \\ s_c &= -(s_n + s_w + s_e + s_s).\end{aligned}$$

with the upwind pressure

$$p^{\uparrow}_{wi} := \begin{cases} p_{wi} & \text{if } p_{wi} > p_{wc} \\ p_{wc} & \text{else.}\end{cases}$$

A possible choice for the non-linear relation K is the van Genuchten-Mualem model. If the material would be heterogeneous, K would be the harmonic mean $K(p_w^{\uparrow}) = (K_c(p_w^{\uparrow})^{-1} + K_n(p_w^{\uparrow})^{-1})^{-1}$. This discretization results in a non-linear system of equations, which has to be solved using a suitable non-linear solver. Typical choices are inexact Newton-Krylov methods, which internally solve a linear system using a Krylov method (e.g. the CG method). In this linear solver, the operator can be evaluated using the above stencil representation. Details regarding the parameterization of the non-linear solver go beyond the scope of this paper.

Summary Based on the equations presented in this section, it can be seen that stencils to solve a given application from the porous-media domain can differ considerably from the commonly known 5-point stencil. For simple equations, such as the heat equation, a simple 5-point stencil can be used if a adequate discretization is selected. In contrast, the stencils for more complex equation, such as Darcy's law, are much more complex. These stencils work, for example, on matrix data, instead of evaluating a scalar value for each entry. Thus, when developing a domain-specific language or a stencil compiler, one has to keep in mind that stencils can look very differently and can become very complex if the problem considered in the application is complex, too. These characteristics of the stencil also have to be considered, when optimizing the performance of an application.

6. RELATED WORK

To the best of our knowledge, there is only little work on the relationship between the characteristics of a partial differential equation and the stencil that can be used to solve the equation after a discretization is performed. However, there is substantial work on optimizing the performance of a specific stencil or multigrid code. For example, machine learning has been used to identify either the optimal configuration or correlations between properties of the code and the performance. For example, Ganapathi et al. uses a kernel canonical correlation analysis to identify the optimal configuration [29]. Another machine-learning approach based on multivariate regression in combination with forward feature selection has been proposed by Siegmund et al. [30].

There is also considerable work on optimization strategies to efficiently parallelize a given stencil computation and to improve the ratio between executed floating-point operations and the bytes fetched from memory. Common strategies are temporal or spatial blocking [31, 32]. Additionally, specific frameworks and methods, such as polyhedral loop optimization, can be used to fully parallelize a given stencil computation automatically [21, 22].

Yu and Smith propose an approach to improve reusability of finite-element analyses using product line technology [33]. In contrast to our work, they focus on the variability of the finite-element analysis to solve beam analysis problems. As a consequence, while we consider general variability that have to be considered during the discretization process of a mathematical model leading to a stencil code for the porous-media domain, they consider variability such as how to calculate the stiffness matrix of the equation.

In a parallel line of research, Smith et al. propose a commonality analysis template for scientific computing of physical models [34]. Using their results, it is possible to see, for example, dependencies between assumptions and the theoretical models. However, in contrast to our work, they focus on the commonalities and differences of mathematical models only, while we include also the requirements on a stencil to solve a given mathematical problem. Furthermore, we emphasize the connection between the stencils and the mathematical models.

7. CONCLUSION

In this article, we discuss the variability of applications in the domain of porous-media flows, and we present a family of models that grows in complexity for this domain. Depending on the characteristics of the mathematical model at hand, different numerical methods have to be used to discretize the underlying equation, and specific stencils are needed to compute the solution. We present two variability models to give an overview of the commonalities and differences of the mathematical models and the resulting stencils. These models can be extended to cover a broader range of the application domain at hand and also to other application domains that result in different partial differential equations. The stencils used to compute approximate solutions in this application domain deviates quite a bit from the standard 5-point stencil used in many benchmarks for stencil codes. To demonstrate the influence of a mathematical model on the resulting stencil, we discuss a set of non-trivial mathematical models together with the stencils used to solve the equations efficiently. The variability models and the relations between the model, the discretization, and the stencils, presented in this article shall help to test advanced compiler technologies and code generators with a wider variety of stencils that are suitable for more demanding and more practical applications than solving the pure heat equation. Additionally, we want to support application engineers in understanding the complexity of stencils and the connection between application, discretization and the stencil.

ACKNOWLEDGEMENTS

This work is supported by the German Research Foundation (DFG), as part of the Priority Programme 1648 "Software for Exascale Computing", under the contracts AP 206/7, BO 3405/2, and EN 1042/2. Sven Apel's work is also supported by the DFG under the contracts AP 206/4 and AP 206/6. This has been cooperative work initiated at the Dagstuhl Seminar *Advanced Stencil-Code Engineering* in April 2015.

REFERENCES

1. Sadourny R. The Dynamics of Finite-Difference Models of the Shallow-Water Equations. *Journal of the Atmospheric Sciences* 1975; **32**(4):680–689.
2. Pletcher, RH and Tannehill, JC and Anderson, D. *Computational Fluid Mechanics and Heat Transfer, Second Edition*. Series in Computational and Physical Processes in Mechanics and Thermal Sciences, Taylor & Francis, 1997.
3. Bear J. *Dynamics of Fluids in Porous Media*. Dover Publications, 1988.
4. Saad Y. *Iterative Methods for Sparse Linear Systems*. 2nd edn., Society for Industrial and Applied Mathematics, 2003.
5. Trottenberg U, Oosterlee CW, Schüller A. *Multigrid*. Academic Press, 2001.
6. Bear J, Bachmat Y. *Introduction to Modeling of Transport Phenomena in Porous Media*, vol. 4. Springer, 2012.
7. Mualem Y. A New Model for Predicting the Hydraulic Conductivity of Unsaturated Porous Media. *Water Resources Research* 1976; **12**:513–522.
8. Van Genuchten MT. A Closed-form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils. *Soil science society of America journal* 1980; **44**(5):892–898.
9. Brooks R, Corey A. *Hydraulic Properties of Porous Media*. Colorado State University Hydrology Papers, Colorado State University, 1964.
10. Celia MA, Russell TF, Herrera I, Ewing RE. An Eulerian-Lagrangian Localized Adjoint Method for the Advection-Diffusion Equation. *Advances in Water Resources* 1990; **13**(4):187–206.
11. Buckingham E. *Studies on the Movement of Soil Moisture*. Bulletin no. 38, Govt. Print. Off., 1907.
12. Van der Vorst HA. *Iterative Krylov Methods for Large Linear Systems*. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2003.
13. Quarteroni A, Sacco R, Saleri F. *Numerical Mathematics*. Texts in Applied Mathematics, Springer, 2000.
14. Knabner P, Angerman L. *Numerical Methods for Elliptic and Parabolic Partial Differential Equations*. Texts in Applied Mathematics, Springer, 2003.
15. Durlofsky LJ. Accuracy of Mixed and Control Volume Finite Element Approximations to Darcy Velocity and Related Quantities. *Water Resources Research* 1994; **30**(4):965–973.
16. Braess D. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, 2001.
17. Wheeler MF. An Elliptic Collocation-Finite Element Method with Interior Penalties. *SIAM Journal on Numerical Analysis* 1978; **15**(1):152–161.
18. Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems. *SIAM Journal on Numerical Analysis* 2002; **39**(5):1749–1779.
19. Apel S, Batory D, Kästner C, Saake G. *Feature-Oriented Software Product Lines: Concepts and Implementation*. Springer, 2013.
20. Kang KC, Cohen SG, Hess JA, Novak WE, Peterson AS. Feature-Oriented Domain Analysis (FODA) Feasibility Study. *Technical Report CMU/SEI-90-TR-2*, CMU, SEI 1990.
21. Ghysels P, Vanroose W. Modeling the Performance of Geometric Multigrid Stencils on Multicore Computer Architectures. *SIAM Journal on Scientific Computing* 2015; **37**(2):C194–C216.

22. Lengauer C, Apel S, Bolten M, Größlinger A, Hannig F, Köstler H, Rüde U, Teich J, Grebhahn A, Kronawitter S, *et al.*. ExaStencils: Advanced Stencil-Code Engineering. *Euro-Par 2014: Parallel Processing Workshops - Euro-Par 2014 International Workshops, 2014, Revised Selected Papers, Part II*, 2014; 553–564.
23. Nguyen A, Satish N, Chhugani J, Kim C, Dubey P. 3.5-D Blocking Optimization for Stencil Computations on Modern CPUs and GPUs. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC, IEEE, 2010; 1–13.
24. Wellein G, Hager G, Zeiser T, Wittmann M, Fehske H. Efficient Temporal Blocking for Stencil Computations by Multicore-Aware Wavefront Parallelization. *Proceedings of Annual IEEE International Computer Software and Applications Conference*, vol. 1, IEEE, 2009; 579–586.
25. Jin G, Endo T, Matsuoka S. A Parallel Optimization Method for Stencil Computation on the Domain that is Bigger than Memory Capacity of GPUs. *IEEE International Conference on Cluster Computing*, CLUSTER, 2013; 1–8.
26. Bastian P, Reichenberger V. Multigrid for Higher Order Discontinuous Galerkin Finite Elements Applied to Groundwater Flow. *Technical Report 2000-37*, SFB 359, Universität Heidelberg 2000.
27. Riviere B, Wheeler MF. Discontinuous Galerkin Methods for Flow and Transport Problems in Porous Media. *Communications in numerical methods in engineering* 2002; **18**(1):63–68.
28. Ern A, Stephansen AF, Zunino P. A Discontinuous Galerkin Method with Weighted Averages for Advection–Diffusion Equations with Locally Small and Anisotropic Diffusivity. *IMA Journal of Numerical Analysis* 2008; **29**(2):235–256.
29. Ganapathi A, Datta K, Fox A, Patterson D. A Case for Machine Learning to Optimize Multicore Performance. *Proceedings of the USENIX Conference on Hot Topics in Parallelism*, HotPar, USENIX Association, 2009; 1–6.
30. Siegmund N, Grebhahn A, Apel S, Kästner C. Performance-influence Models for Highly Configurable Systems. *Proceedings of the Joint Meeting on Foundations of Software Engineering*, ESEC/FSE, ACM, 2015; 284–294.
31. Treibig J, Wellein G, Hager G. Efficient Multicore-Aware Parallelization Strategies for Iterative Stencil Computations. *J. Computational Science* 2011; **2**(2):130–137.
32. Wittmann M, Hager G, Eitzinger J, Wellein G. Leveraging Shared Caches for Parallel Temporal Blocking of Stencil Codes on Multicore Processors and Clusters. *Parallel Processing Letters* 2010; **20**(4):359–376.
33. Yu W, Smith S. Reusability of FEA Software: A Program Family Approach. *Proceedings of the International Workshop on Software Engineering for Computational Science and Engineering*, SECSE, 2009; 43–50.
34. Smith WS, Carette J, McCutchan J. Commonality Analysis of Families of Physical Models for use in Scientific Computing. *Proceedings of the International Workshop on Software Engineering for Computational Science and Engineering*, SECSE, 2008.

A. DESCRIPTION OF THE FEATURES

Table I. Description of the features of the mathematical part of the porous media domain (cf. Figure 1).

Feature	Description
Continious Galerkin	This finite-element discretization method uses functions that are assumed to be continuous at the element boundaries.
Mixed FEM	This finite-element discretization method introduces independent variables during the discretization.
Discontinuous Galerkin	This finite-element discretization method uses functions being only piecewise continuous.
Finite Volume	The finite-volume discretization starts form an integral formulation of the application.
Finite Differences	The finite-differences discretization seeks a solution at the vertices of the grid using, for example, a differences quotient.
2D	The application is defined over a two dimensional domain.
3D	The application is defined over a three dimensional domain.
Dirichlet.Homogeneous	This boundary condition states that the solution of the equation is prescribed to 0 at the boundary of the domain.
Dirichlet.Inhomogeneous	This boundary condition states that the solution of the equation is prescribed to a value different from 0 at the boundary of the domain.
Periodic	This boundary condition states that the boundary of the domain is directly linked to the opposide side of the boundary.
Neumann.Homogeneous	This boundary condition states that the flux of the equation is prescribed to 0 at the boundary of the domain.
Neumann.Inhomogeneous	This boundary condition states that the flux of the equation is prescribed to a value different from 0 at the boundary of the domain.
Robin	This boundary condition is a weighted combination of the Dirichlet and Neumann boundary conditions.
Outflow	This boundary condition describes the flow out of the domain.
Single Component	The transport of one solute is considered in the equation.
Multi Component	The transport of multiple solutes is considered in the equation.
Single	One fluid that fills the porous medium is considered in the equation.
Multi	Multiple, eventually interacting fluids are considered in the equation.
Homogeneous	States that all coefficients of the domain are homogeneous. As a result, the coefficients do not change over time and space.
Heterogeneous	States that the coefficients are non-constant in the computation. This is because of heterogeneous in the porous media or because of time dependencies. However, in this work, we do not consider time-dependent problems.

Table II. Description of the features of stencils of the domain of the porous media domain (cf. Figure 2).

Feature	Description
Constant	The values of a stencil are constant.
Function	It is necessary to evaluate a function at each point of the computation domain for each entry of the stencil.
Entry Type.Scalar	Defines that one entry of the stencil is scalar, which can be a constant or a function.
Matrix	Defines that one entry of the stencil is a matrix that has to be evaluated in each step.
2D	The stencil is defined over a two dimensional domain.
3D	The stencil is defined over a three dimensional domain.
Halo	The boundaries of the domain are handled by creating a halo region around the domain.
Special Stencil	Special stencils have to be used at the boundaries of the computation domain.
5-Point	A stencil with 5 entries.
9-Point	A stencil with 9 entries.
7-Point	A stencil with 7 entries.
27-Point	A stencil with 27 entries.
Arbitrary	Stencils with a complex shape, that does not correspond to the 4 shapes listed previous.
Data.Scalar	The entries of the stencil only have one dimension.
Vector	Each entry of the stencil is a vector, where each element of the vector can be a constant, a function, or a matrix, depending on its entry type and value type.