Computers and Geosciences 113 (2018) 70–80

Contents lists available at ScienceDirect

Computers and Geosciences

journal homepage: www.elsevier.com/locate/cageo

Research paper

Metamodeling-based approach for risk assessment and cost estimation: Application to geological carbon sequestration planning

Alexander Y. Sun [a,*], Hoonyoung Jeong [a,**,1], Ana González-Nicolás [b], Thomas C. Templeton [a]

[a] Bureau of Economic Geology, Jackson School of Geosciences, The University of Texas at Austin, Austin, TX, 78738, USA
[b] Earth and Environmental Sciences, Lawrence Berkeley National Laboratory, Berkeley, CA, 94720, USA

ARTICLE INFO

Keywords:
Metamodeling
Gaussian process regression
Polynomial chaos expansion
Carbon capture and storage
Risk assessment
Leakage liability
Cloud computing
High performance computing

ABSTRACT

Carbon capture and storage (CCS) is being evaluated globally as a geoengineering measure for significantly reducing greenhouse emission. However, long-term liability associated with potential leakage from these geologic repositories is perceived as a main barrier of entry to site operators. Risk quantification and impact assessment help CCS operators to screen candidate sites for suitability of CO_2 storage. Leakage risks are highly site dependent, and a quantitative understanding and categorization of these risks can only be made possible through broad participation and deliberation of stakeholders, with the use of site-specific, process-based models as the decision basis. Online decision making, however, requires that scenarios be run in real time. In this work, a Python based, Leakage Assessment and Cost Estimation (PyLACE) web application was developed for quantifying financial risks associated with potential leakage from geologic carbon sequestration sites. PyLACE aims to assist a collaborative, analytic-deliberative decision making processes by automating metamodel creation, knowledge sharing, and online collaboration. In PyLACE, metamodeling, which is a process of developing faster-to-run surrogates of process-level models, is enabled using a special stochastic response surface method and the Gaussian process regression. Both methods allow consideration of model parameter uncertainties and the use of that information to generate confidence intervals on model outputs. Training of the metamodels is delegated to a high performance computing cluster and is orchestrated by a set of asynchronous job scheduling tools for job submission and result retrieval. As a case study, workflow and main features of PyLACE are demonstrated using a multilayer, carbon storage model.

1. Introduction

Fossil fuels are expected to continue to supply a large part of the world's energy needs (86 percent in 2015) in the near future (BP, 2016). Carbon capture and sequestration (CCS) is the process of capturing CO_2 from power plants and industrial emission sources and then injecting into a deep subsurface repository, usually a saline aquifer or depleted oil/gas reservoir, for permanent storage and sequestration. CCS offers a potential solution for tapping into the energy benefits of fossil energy while mitigating anthropogenic greenhouse gas emissions (Haszeldine, 2009). Main technical challenges pertaining to all geologic carbon sequestration operations are to ensure that injectivity and capacity predictions are reliable, the fate of the injected CO_2 plume is accountable, and anomalies related to unexpected fluid migration are detectable (Harbert et al., 2016). Over the past decade, the CCS community has significantly advanced and enriched its knowledge base through extensive modeling, laboratory, and field demonstration projects (Birkholzer et al., 2015; Jenkins et al., 2015). A recent survey of 229 CCS experts showed that most participants shared broad confidence in the readiness of CCS for commercialization; nevertheless, surveyees also identified four major barriers to CCS commercialization, including (1) cost and cost recovery, (2) lack of a price signal or financial incentive, (3) lack of a comprehensive regulatory regime, and (4) long-term liability risks (Davies et al., 2013). Unlike the first three barriers that are related to exogenous policy and market drivers, the long-term liability risk is largely endogenous and related to the fact that the probability of leakage via natural (geologic faults and fractures) and manmade (e.g., abandoned wells) pathways in a geologic repository is a priori nonzero (Wilson et al., 2007; Pollak and

* Corresponding author.
** Corresponding author.
E-mail addresses: alex.sun@beg.utexas.edu (A.Y. Sun), dragory@gmail.com (H. Jeong).
1 Now with Department of Energy Resources Engineering, College of Engineering, Seoul National University, 599 Gwanak-gu, Seoul 08826, Republic of Korea.

https://doi.org/10.1016/j.cageo.2018.01.006
Received 10 May 2017; Received in revised form 11 December 2017; Accepted 14 January 2018
Available online 31 January 2018
0098-3004/

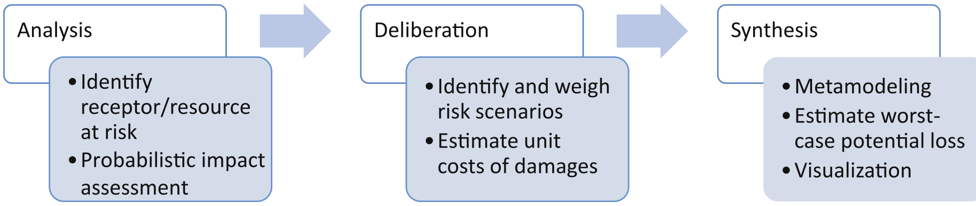

Fig. 1. A decision support workflow for CCS.

McCoy, 2011).

Leakage risk assessment has two major components: risk quantification and impact analysis. For the former component, extensive studies have been conducted on process-level understanding of leakage pathways and mechanisms (Benson and Cole, 2008; Carroll et al., 2009; Birkholzer et al., 2011), quantification of potential leakage risks (Viswanathan et al., 2008; Sun et al., 2013b; Pawar et al., 2015), and leakage detection technologies (Oldenburg et al., 2009; Jenkins et al., 2015; Sun et al., 2013a; Harbert et al., 2016). Under the U.S. Department of Energy's National Risk Assessment Partnership (NRAP) program, a suite of integrated assessment models were developed using system modeling approaches to quantify risks and risk profiles for generic carbon sequestration sites (Pawar et al., 2015). On the other hand, the CCS impact analysis has received relatively limited attention, mainly because impact and liability are often poorly defined, and multiple stakeholders and the public tend to "use the terms risk, injury, damage, liability, and financial responsibility differently than each other and sometimes interchangeably" (Wilson et al., 2003, 2007; Trabucchi et al., 2010; Bielicki et al., 2014). Elsewhere in environmental and natural resources management, various decision support systems (DSS) have long been used to engage stakeholder deliberation and to build consensus around different management actions (Matthies et al., 2007; Rehr et al., 2012; Sun, 2013). The democratization of environmental policy decisions is often referred to as an analytic-deliberative process combining analysis (input from the physical and social sciences) and deliberation (input from stakeholders) (NRC, 1996). In many situations, such collaborative analytic-deliberative process constitutes the key to fruitful decision makings and can often lead to widely accepted risk policies (Arvai, 2003; Rehr et al., 2012; Uusitalo et al., 2015). Along the same vein, the CCS community may also benefit from DSS that link integrated assessment modeling with participatory impact analyses, such that concrete assessments of the potential costs and compensatory damages related to site-specific CCS activities can be performed systematically, collaboratively, and transparently.

We envision that a CCS-oriented decision making process may include three key stages: analysis, deliberation, and synthesis (Fig. 1). During the analysis stage, the first step is to characterize site-specific risk pathways or damage categories that may incur financial and economic consequences, including, but not limited to, groundwater contamination, induced seismicity, surface/subsurface mineral right infringement and asset damage, and property damage resulting from leakage or ground deformation (Trabucchi et al., 2010). Outcomes from the first step shall provide inputs to the second step, in which a probabilistic impact assessment of different risk scenarios will be conducted.

In the deliberation stage, outputs from the analysis stage are assessed, and damage costs, including environmental remediation and legal costs, are derived. Cost attribution is arguably the most challenging step because of the lack of leakage cost databases in the literature. Although knowledge gained from oil and gas industry may be used as proxies for appraising costs associated with leakage (Wilson et al., 2003; Jordan and Benson, 2009), a more constructive approach is to solicit inputs from stakeholders and domain experts that are directly involved in a particular CCS project. A recent application of the latter approach is presented in Bielicki et al. (2014), in which low- and high-cost story lines (i.e., narratives of leakage outcomes) are formed by surveying stakeholders who may be potentially affected by leakage events. Thus, similar to many other environmental decision making problems, the deliberation process is implicitly participatory and iterative, calling for an effective communication platform that may help to build trust and transparency among decision actors.

In the synthesis stage, outputs from the first two stages are combined to assess potential losses expected at a CCS site. For planning purposes, common tasks include estimation of the maximum potential loss (e.g., worst-case scenario), and identification of the most significant system parameters that could affect the expected losses. The synthesis process is similar to design of experiments in the sense that it is also iterative, requiring exploration of the decision space to find suitable design parameters and then estimating the range of expected losses. Such a task can be computationally demanding if process-level models are used directly and the dimension of decision space is relatively large. Metamodeling, which is concerned with developing a faster-to-run surrogate of the process-level model(s), offers a potential solution for accomplishing the aforementioned risk assessment tasks online, reducing a significant gap between decision analysis and deliberation processes. Metamodeling has been a subject of intensive studies in recent years (Sudret, 2008; Villa-Vialaneix et al., 2012; Razavi et al., 2012; Maier et al., 2014; Sun and Sun, 2015; Sun et al., 2015), however, few have addressed the use and deployment of metamodels in an online environment for supporting decision making.

The main objective of this work is thus to develop and demonstrate a Python-based, Leakage Assessment and Cost Estimation (PyLACE) web application, with an emphasis on facilitating the transformation of process-level analysis models to support online probabilistic impact analyses. Web-based platforms are highly desirable for integrated assessment and decision making, providing a common base for team interactions and deliberation, as suggested by many recent studies (Buytaert et al., 2012; Castronova et al., 2013; Laniak et al., 2013; Sun, 2013; Vitolo et al., 2015; Swain et al., 2016). In particular, the way through which an individual or a group of people receive a particular type of information can radically influence whether the information sticks and gets passed on (Gladwell, 2006). The fast evolution of cyberinfrastructure and social media in recent years have dramatically changed the landscape of scientific computing and team collaboration, making it more feasible and affordable than ever to perform distributed modeling in the Cloud (i.e., software as a service) and to develop and deploy web solutions by using user-defined computing technologies and operating systems (i.e., infrastructure as a service) (Goodall et al., 2011; Sun, 2013; Rajib et al., 2016).

For the purpose of this work, we assume that the user has already developed a process-level model and identified a set of most important uncertain system parameters through, for example, global sensitivity analysis (Saltelli et al., 2008). Further, we assume that the process-level model has already been approved by the stakeholders involved with the project. Thus, main functionalities of PyLACE are to (1) orchestrate the creation of metamodels from a user-provided process-level model, (2) perform probabilistic impact analysis using the generated metamodel,

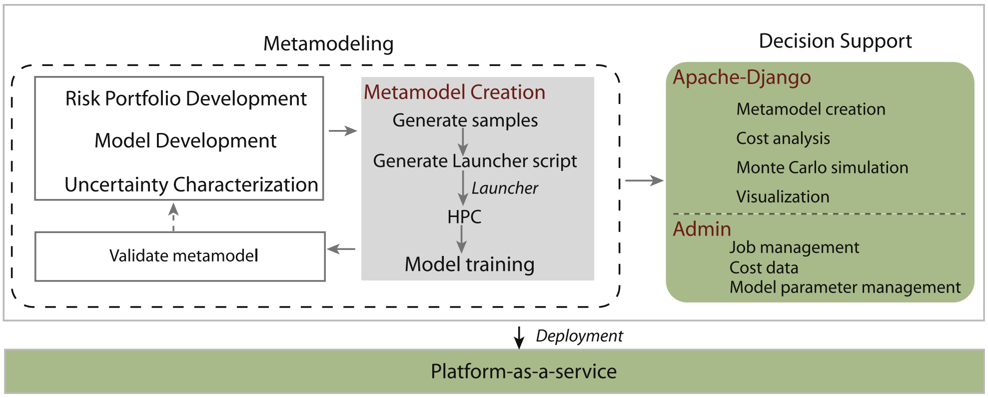

Fig. 2. Architectural diagram of PyLACE, which consists two main functional blocks, metamodel development (left box with dashed line border) and metamodel-based decision support (right box shaded in green). PyLACE provides functions to facilitate interactions among the user, the metamodeling creation process, and high-performance computing (HPC). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

and (3) enable multi-stakeholder deliberation of the potential impact or liability. The entire framework is implemented using open-source packages, most of which are written in Python. In the following, Section 2 describes the system design and metamodeling methodologies, Section 3 demonstrates the tool usage through a use case study. Finally, conclusions and findings are given in Section 4.

2. Methodology

Fig. 2 shows the overall architecture of PyLACE, which includes two major functional blocks: metamodeling (left block in dashed line), and web-based probabilistic risk assessment and cost estimation (right block in green color). Each block, in turn, consists of multiple modules. By design, the modules are loosely coupled, communicating with each other through messaging (e.g., JavaScript Object Notation or JSON). Thus, technologies behind each module can be easily expanded or tailored according to users' needs. Below we provide technical details of these blocks and their implementation.

2.1. Metamodeling

Any process-level model for predicting the movement and fate of injected CO_2 plume can be subject to uncertainties in model conceptualization (model structure), model parameters, and initial and boundary conditions. Let such an uncertain, process-level model be denoted as

$$u = f(\mathbf{v}, \boldsymbol{\theta}), \tag{1}$$

where $u = u(\mathbf{x}, t)$ is model solution, f is forward model, $\mathbf{v}$ is a set of deterministic model parameters, and $\boldsymbol{\theta}$ is a set of uncertain model parameters including both scalars and distributed variables. The idea of metamodeling is to find a surrogate model $\hat{f}$ that converges to f in the mean square sense, but is computationally cheaper than the original model (Sun and Sun, 2015). Commonly used metamodeling techniques are originated from machine learning literature, including, for example, artificial neural network, polynomial regression, Gaussian process regression (GPR), kriging, multivariate adaptive regression splines, decision tree, and support vector machine (Bishop, 2006; Wang and Shan, 2007; Villa-Vialaneix et al., 2012; Sun and Sun, 2015). Currently, PyLACE supports two metamodeling algorithms, a special type of stochastic response surface method and GPR, both are highly suitable for the task at hand based on our previous experience and both allow for evaluation of input uncertainty (Sun et al., 2013b, 2015).

2.1.1. A stochastic response method

Various stochastic response surface methods have been used for model approximation and uncertainty quantification in porous media modeling (e.g., Li and Zhang, 2007; Zeng et al., 2012; Zhang et al., 2013). In PyLACE, a special stochastic response surface method based on the polynomial chaos expansion (PCE) theory (Xiu and Karniadakis, 2002; Ghanem and Spanos, 2003) is implemented.

The PCE, which is based on the homogeneous chaos theory by Wiener (1938), provides a stochastic representation of random variables. Here PCE is used to expresses the model output u as a projection of input random variables onto a stochastic space spanned by a set of orthogonal polynomials $\{\Phi_i\}$,

$$u \approx \hat{f}(\mathbf{x}, t) = \sum_{i=0}^{N_p} f_i(\mathbf{x}, t)\Phi_i(\boldsymbol{\xi}), \tag{2}$$

in which $\{f_i\}$ is a set of deterministic coefficients, N_p is the order of expansion, and $\boldsymbol{\xi} = \{\xi_1, \xi_2, \ldots, \xi_N\}$ is a set of N univariate random variables. For clarity, we shall omit the dependency on $(\mathbf{x}, t)$ in the following presentation where no confusion should occur. The basis functions $\{\Phi_i\}$ are multidimensional polynomials satisfying orthogonality constraint

$$\left\langle \Phi_i, \Phi_j \right\rangle = \int \ldots \int_{\Omega} \Phi_i(\boldsymbol{\xi})\Phi_j(\boldsymbol{\xi})p(\boldsymbol{\xi})d\boldsymbol{\xi} = \langle \Phi_i \rangle^2 \delta_{ij}, \tag{3}$$

in which $p(\boldsymbol{\xi})$ is joint probability distribution of $\boldsymbol{\xi}$, δ_{ij} is Kronecker operator which is 1 when $i = j$ and 0 otherwise, and $\langle \cdot \rangle$ is an expectation operator over the stochastic space Ω. The total number of terms in PCE is determined by the random dimension N and the highest-order, p, of the orthogonal polynomials used in the expansion

$$N_p + 1 = \frac{(N+p)!}{N!p!}. \tag{4}$$

Eq. (4) indicates that the number of terms grows exponentially with the random dimension, suggesting PCE is most useful for low-dimensional problems. However, such a "curse-of-dimensionality" issue is not unique to PCE-based metamodeling. Thus, an important step during the "risk portfolio development" stage in Fig. 2 is to identify a small set of most influential parameters for uncertainty quantification and leave the rest of the parameters as deterministic variables.

Depending on the type of distribution, different orthogonal polynomials need to be used to ensure optimality. The most commonly used method for constructing $\{\Phi_i\}$ is the Askey-Wilson scheme (Askey and Wilson, 1985), which specifies the optimal orthogonal polynomials for a large number of probability distributions (e.g., Hermite polynomial for Gaussian distributions and Legendre polynomial for uniform distributions). Determination of the coefficients $\{f_i\}$ is usually the most computationally demanding step. For this purpose, two categories of methods exist, the intrusive and non-intrusive methods, with the main difference being that the former requires modification of the numerical solver. Thus, non-intrusive methods are often preferred over intrusive methods, especially in black-box applications. Non-intrusive methods can be further divided into integration methods and point collocation methods. Integration methods obtain the coefficients $\{f_i\}$ by numerically integrating a multidimensional integral using a quadrature rule. On the other hand, point collocation methods first obtain model outputs at a set of predefined points (collocation points), using which an over-determined system of equations is formed and then solved using linear regression to obtain the coefficients. In either case, a large number of runs of the original model is required. Generating random samples (collocation points) from the joint probability distribution of $\boldsymbol{\xi}$ may not be trivial. A common approach is to assume that all components are independent random variables so that only univariate sampling is required. Correlated random fields may be approximated by a linear combination of independent random variables using, for example, Karhunen-Loève transform (Ghanem and Spanos, 2003). Multivariate sampling techniques, such as Rosenblatt transform (Rosenblatt, 1952), have been implemented to allow sampling from arbitrary distributions.

Application of a PCE-based metamodeling technique can be summarized in four main steps: (1) specification of the uncertain variables and their distributions, (2) construction of orthogonal polynomials, and generation of nodes and weights (if integration method is used) or collocation points (if point collocation method is used), (3) calculation of model outputs at the nodes, and (4) formulation of a metamodel for online simulation and uncertainty quantification. In this work, the open-source Python toolbox, Chaospy, developed by Feinberg and Langtangen (2015), was used. Chaospy currently supports (1) construction of orthogonal polynomials $\{\Phi_i\}$, for either independent random variables or dependent variables; (2) sampling of a joint distribution of all random variables; and (3) calculation of coefficients $\{f_i\}$ through either integration methods or point collocation methods.

2.1.2. Gaussian process regression (GPR)

GPR represents another commonly used metamodeling methodology for estimating a mapping between training points (in multidimensional parameter space) and discrete function values. It has been applied in water distribution network design (Krause and Guestrin, 2007), contaminant source identification (Zhang et al., 2016), streamflow forecasting (Sun et al., 2014), and atmospheric modeling (Marrel et al., 2008).

In GPR, the function outputs are treated as realizations of a random process and, in particular, a Gaussian process (GP), that is completely specified by its mean and covariance (Rasmussen, 2006). The starting point of GPR is the following additive error model that is assumed between outputs of a process-level model and its surrogate, $\hat{f}(\boldsymbol{\theta})$

$$u_i(\boldsymbol{\theta}_i) = \hat{f}_i(\boldsymbol{\theta}_i) + \varepsilon_i, \varepsilon_i \in \mathcal{N}\left(0, \sigma_i^2\right), i = 1, 2, \ldots, N_T \tag{5}$$

where ε_i is zero-mean Gaussian random error with variance σ_i^2; N_T is the total number of training points $\{\boldsymbol{\theta}_i\}$; and $\{u_i\}$ are the corresponding process-level model outputs. Denoting $\mathbf{u} = \{u_i\}$, $\Theta = \{\boldsymbol{\theta}_i\}$, $\mathbf{f} = \{\hat{f}_i\}$, and $\mathbf{z} = \{\varepsilon_i\}$, Eq. (5) may be written in a vector form as

$$\mathbf{u}(\Theta) = \mathbf{f}(\Theta) + \mathbf{z}(\Theta) \tag{6}$$

Because of the GP assumption, the prior PDF of $\mathbf{f}$ is Gaussian

$$p(\mathbf{f}|\boldsymbol{\beta}) \propto \mathcal{N}(0, \mathbf{K}), \tag{7}$$

where $\mathbf{K}$ is an $N_T \times N_T$ covariance matrix with kernel function elements $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$;$\boldsymbol{\beta}$ are parameters used to define the functional form of $\hat{f}(\boldsymbol{\theta})$, as well as parameters of the kernel function (see below). The most commonly used form of $\hat{f}(\boldsymbol{\theta})$ is a linear combination of basis functions, similar to that given in Eq. (1),

$$\hat{f}(\boldsymbol{\theta}_i) = \sum_{j=1}^{M} w_j \phi_j(\boldsymbol{\theta}_i) = \mathbf{w}^T \phi(\boldsymbol{\theta}_i), i = 1, 2, \ldots, N_T \tag{8}$$

where $\mathbf{w} = (w_1, \ldots, w_M)^T$ is a weight vector and $\{\phi_j\}$ is a set of basis functions. Applying Eq. (8) to all discrete values in $\mathbf{f}$ leads to

$$\mathbf{f} = \mathbf{\Phi}\mathbf{w}, \tag{9}$$

in which the $N_T \times M$ matrix $\mathbf{\Phi}$ is called the design matrix, with each row containing the outputs of all basis functions calculated on a specific training sample, namely,

$$\phi_i = [\phi_1(\boldsymbol{\theta}_i), \phi_2(\boldsymbol{\theta}_i), \ldots, \phi_M(\boldsymbol{\theta}_i)], i = 1, \ldots, N_T \tag{10}$$

It can be shown that the relationship between covariance matrix $\mathbf{K}$ and design matrix $\mathbf{\Phi}$ is (Rasmussen, 2006)

$$\mathbf{K} = \mathbf{\Phi}\mathbf{\Sigma}_w\mathbf{\Phi}^T, \tag{11}$$

where $\mathbf{\Sigma}_w$ is the covariance matrix of weight vector $\mathbf{w}$. The likelihood and posterior PDF of $\mathbf{f}$ are also Gaussian, and the mean and covariance of the posterior PDF are given by (Rasmussen, 2006)

$$\boldsymbol{\mu} = \mathbf{K}^T(\mathbf{K} + \mathbf{\Sigma}_z)^{-1}\mathbf{u}, \tag{12}$$

$$\mathbf{\Sigma} = \mathbf{K} - \mathbf{K}^T(\mathbf{K} + \mathbf{\Sigma}_z)^{-1}\mathbf{K}, \tag{13}$$

where $\mathbf{\Sigma}_z$ is model error covariance matrix. Note that only the kernel function is required to calculate mean and covariance, but not the actual forms of basis functions, nor weights $\mathbf{w}$. This is known as the "kernel trick" in machine learning (Bishop, 2006). PyLACE adopts a radial-basis function kernel defined as

$$k(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sigma_f^2 \exp\left(-\frac{r^2}{2l^2}\right) \tag{14}$$

where $r = ||\boldsymbol{\theta} - \boldsymbol{\theta}'||$ is the Euclidean distance between two input points $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, and length scale l and variance σ_f^2 are hyperparameters. Training of the GPR then entails finding the values of the hyperparameters and the variance of model error, σ^2, by minimizing a log marginal likelihood function (Rasmussen, 2006). We use the Python package GPy (GPy, 2012) to train GPR models.

2.2. Leakage cost estimate

Leakage cost estimate is an integral part of CCS project planning. In general, local stakeholder concerns and leakage risks dominate during the CCS planning stage. Issues such as the ownership of injected CO_2, damage attribution and partition among multiple actors, and remediation liability must be addressed in a site-specific manner (Wilson et al., 2007). The outcome of liability cost analysis may serve several purposes, for instance, for securing an insurance policy or transferring liability from private firms to a community pool (e.g., trust fund). In their case study related to a CCS project in Michigan sedimentary basin, Bielicki et al. (2014) solicited inputs from local experts, oil and gas engineers, regulators, academics, attorneys, and other environmental professionals who were knowledgeable about the basin to estimate costs related to (1) finding and fixing leak, (2) environmental remediation, (3) injection interruption, (4) legal costs, and (5) business disruptions to others. They found that the majority of leakage costs is incurred from activities related to finding and fixing a leak and to injection interruptions.

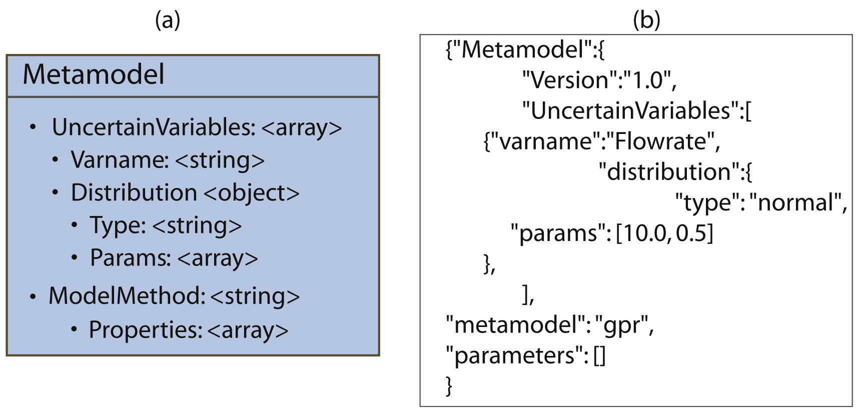

Fig. 3. (a) JSON schema used in PyLACE for specifying uncertain variables and metamodeling algorithm, in which <> indicates data types; (b) an example of JSON file that specifies a single uncertain variable (flow rate) and its distribution information.

To assist stakeholders in their deliberation of potential leakage costs, PyLACE provides a graphic user interface for users to enter cost categories, the weight (or likelihood) of each category, as well as the associated unit costs (e.g., in dollar per ton of leaked fluid). A survey may also be set up using PyLACE to elicit expert opinions on the likelihood of each category. Expert elicitation is a "formal heuristic process of acquiring an expression of the opinion of one or more experts in the form of words, numbers, language, pictures or figures" (Ayyub, 2001). A large number of elicitation techniques are available, as reviewed by Krueger et al. (2012) and the references therein. Creation of a shared and accessible knowledge base is important for improving transparency and unbiasedness of an elicitation process (Krueger et al., 2012). In this work, experts are asked to provide a probabilistic scale of each category, with 1 the least likely, and 10 the most likely. The weight assigned to each category is a weighted arithmetic average of all expert inputs.

The unit damage cost is calculated as the weighted mean of all damage categories. Most of the previous studies investigating the leakage impact considered a CO_2-rich brine as the primary form of leakage (Carroll et al., 2009; Navarre-Sitchler et al., 2013). Here, we allow the user to separate the leakage impact assessment of brine and CO_2-rich brine based on the following rationales: (1) the footprint of pressure perturbation caused by injection is generally much greater than that of CO_2 plume (Zhou et al., 2008); (2) leakage would be dominated by brine leakage before CO_2 plume arrival; (3) chemical reactions with and without dissolved CO_2 are different; (4) brine leakage is as detrimental as CO_2 leakage, if not worse, because of its high concentrations of heavy metals; and (5) unexpected CO_2 migration to neighboring properties may disrupt their operations because of the acidic nature of CO_2 gas and potential corrosion (Islam and Sun, 2016), while the oil production infrastructure is generally more tolerant to brine intrusion by design. Thus, it is desirable for the metamodels to provide estimates of damage caused by leaked CO_2 and brine separately.

2.3. System implementation

Features currently supported by PyLACE are represented in the shaded blocks in Fig. 2. The web interface is implemented on top of an open-source Python web framework, Django (Django Project, 2017), which supports a model-view-controller web design pattern. Thus, Django is particularly useful for developing data-driven web applications, in which the programmer specifies business logics and data abstraction (model), the control of user interactions (controller), and the look-and-feel of the frontend (view). Django has been used in several environmental web-GIS applications (Sun, 2013; Swain et al., 2016).

PyLACE allows the user to specify uncertain variables and metamodeling algorithm in JSON format, which is a standard file format for web application messaging and can be easily edited by using any text editor. Fig. 3a shows the JSON schema used in PyLACE, in which the main tag groups are UncertainVariables and metamodel. Currently available metamodel choices are either pce or gpr. For the former model, the parameter is the order of polynomials to be used, while for the latter there is no required parameter.

An example implementation of the JSON schema is given in Fig. 3b, in which only one uncertain variable (flowrate) is included, which is normally distributed with mean (10.0) and standard deviation (0.5), as specified under the params tag.

A typical workflow may involve the following two steps. First, the user uploads a JSON metamodel specification file via PyLACE's

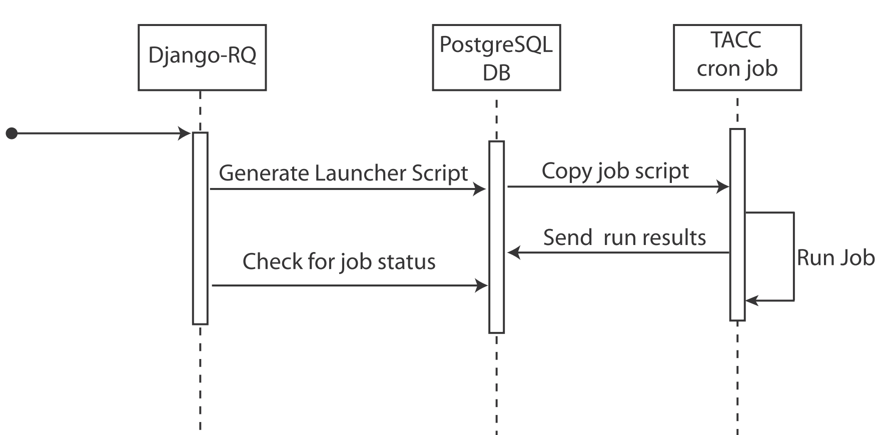

Fig. 4. Sequence diagram of metamodeling job management.

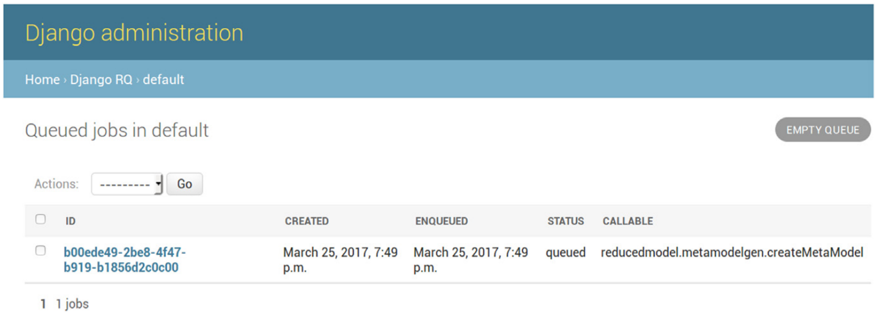

Fig. 5. Submitted jobs can be managed and monitored using Django admin dashboard.

metamodeling module. Second, the user launches a process to create a metamodel. At this point, several actions happen depending on the algorithm specified in the metamodel file. For PCE solver, collocation points are created by sampling from the joint PDF of all random variables, according to the polynomial order specified by the user in the JSON file. For GPR solver, a large number of training points are sampled from the joint PDF using Latin hypercube sampling. Note the actual number of training points required is problem dependent. Validation tests can be performed to check the accuracy of the trained models. Recently, adaptive methods (or exploration-exploitation) have also been used to train GPR, in which the training points are added gradually until certain model performance criterion is reached (Desautels et al., 2014; Zhang et al., 2016). PyLACE then translates the generated collocation point set or training set into a launcher job script. Launcher is a distributed computing utility developed by the Texas Advanced Computing Center (TACC) at the University of Texas. It is used to run a large number of serial or multi-threaded applications as a single, multinode parallel job on HPC clusters (Wilson and Fonner, 2014).

Because all tasks involved in metamodel creation are potentially time consuming, PyLACE uses a Django-based job management tool, django-RQ (https://github.com/ui/django-rq), to schedule the jobs as asynchronous tasks. Fig. 4 shows a sequence diagram of these different tasks. During run time, a cron job scheduled on the TACC cluster constantly checks for new launcher jobs in a PostgreSQL database deployed on the web server side. This is necessary for this study because TACC implements a multi-factor authentication for login, making it easier to pull jobs from inside TACC resources than to push jobs to TACC from the web server side. When a new launcher job is found in the database, the cron job copies the job script, submits the job to the HPC, and periodically checks for the job completion status. After the HPC job is finished, the cron job copies the results back to the web server. The PostgreSQL database is used as a intermediate place for different tasks to update task status. The user may monitor job status using Django's admin interface (see Fig. 5). Email notification can also be added. Although a large part of the web server and HPC interactions described here pertains to the particular HPC used for this study, the methodology is general and may be readily applied to any batch-scheduled HPC environment.

The PyLACE application was developed on a virtual machine running CentOS (https://www.centos.org/) Linux operation system. The Apache (https://www.apache.org/) web server is used to host Django applications. Because of the large number of the Python package used, a Python package manager, fabric (http://www.fabfile.org), is used to automate

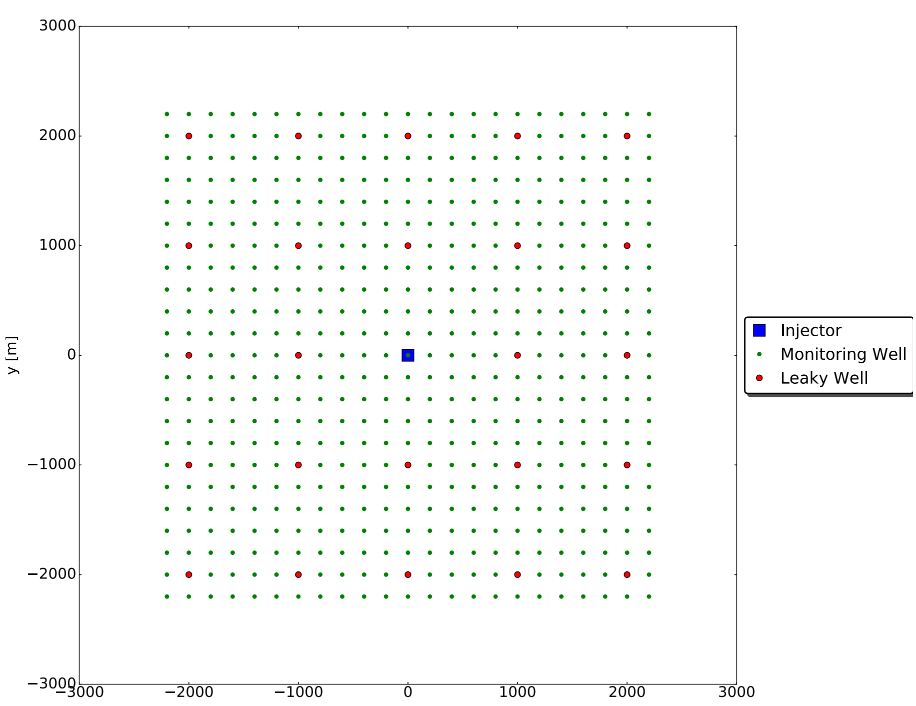

Fig. 6. Plan view of the injection layer, which shows locations of leaky wells (red dots), observation wells (green dots), and a single injector (blue square). Note in this example the leaky and observation wells should be interpreted as virtual wells, where either leakage impact assessment or model outputs are needed for worst-case assessment. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Deterministic parameters used in the multi-layer CO_2 leakage model.

Parameter	Value [unit]	Parameter	Value [unit]
CO_2 density	479 [kg/m^3]	CO_2 viscosity	3.95×10^{-5} [Pa·S]
Brine density	1045 [kg/m^3]	Brine viscosity	2.535×10^{-4} [Pa·S]
Total compressibility	4.2×10^{-10} [1/Pa]	Brine residual saturation	0.2

application deployment and manage Python package dependencies. Alternatively, the virtual machine image may be directly uploaded to a Cloud platform.

3. Demonstration

3.1. Description of the use case

Here the workflow of PyLACE is demonstrated using a synthetic scenario, in which CO_2 is injected into a multi-layer storage site consisting of an injection zone (reservoir), a confining unit (aquitard), and a number of above-zone aquifers. All aquifers are horizontal, homogeneous, and confined. The reservoir may have one or more injectors, and one or more abandoned wells that are potentially leaky. This stylized, stacked-aquifer configuration has been used extensively in the CCS research community for benchmarking purposes. Specifically, models based on such conceptualization are behind several risk assessment tools, such as ELSA (Celia et al., 2011; Bielicki et al., 2014; González-Nicolás et al., 2015) and CO_2-PENS (Stauffer et al., 2006).

Assuming (1) horizontal flows in the aquifers, (2) capillary pressure is negligible, (3) sharp fluid interface between brine and CO_2, and (4) pressure responses from sources and sinks can be super-imposed on each aquifer, a semi-analytical solution can be obtained by using Green's function as a fundamental solution and solving for unknown pressures at observation locations (see Appendix A) (Nordbotten et al., 2009). Recently, an iterative global pressure solution procedure was also proposed to speed up calculation by eliminating the need to solve a large set of linear equations at each time step (Baù et al., 2015). Nevertheless, the computational effort may still be significant when the number of leaky wells and/or model output locations (i.e., virtual observation locations) is large.

As an illustration, a three-layer system consisting of an injection zone, aquitard, and single above-zone aquifer is considered. The region of interest is 4000 m×4000 m. The whole field has 24 abandoned wells, which are uniformly distributed. The underlying assumption is that the probability of leakage is uniform across the field and the worst-case scenario is desired for leakage impact assessment. In other words, the leaky well locations may not correspond to actual wells; instead, they represent locations for impact assessment. All leaky wells penetrate the three formations and the well permeability is set to 1×10^{-12} m^2 (1 darcy). Fig. 6 shows a plan view of the model domain, where locations of injector, leaky wells, and observation wells are labeled. Model outputs are calculated at "virtual" observation well locations, with a uniform spacing of 200 m. The deterministic model parameters are given in Table 1. The total injection period is 20 years.

Fig. 7 shows a screenshot of an uploaded metamodel file that specifies the uncertain parameters and their statistics (mean and variance of log-normal distributions, and lower- and upper-bound for uniform distributions). In this case, the uncertain variables include reservoir and aquifer permeability (m^2), aquitard thickness (m), reservoir and aquifer porosity (-), and injection rate (Mt/year). Their probability distribution types are

Table 2
Damage categories, unit costs, and the corresponding weights.

Type	Category	Unit Cost ($/ton)	Weight
CO_2	Fixing well cost	0.5	0.3
	Legal costs	0.5	0.1
	Environmental remediation	0.5	0.2
	Injection interruption	1.1	0.35
	Business activity interference	2.0	0.05
Brine	Fixing well cost	0.5	0.5
	Legal costs	0.5	0.1
	Injection interruption	1.5	0.3
	Groundwater remediation	1.0	0.1

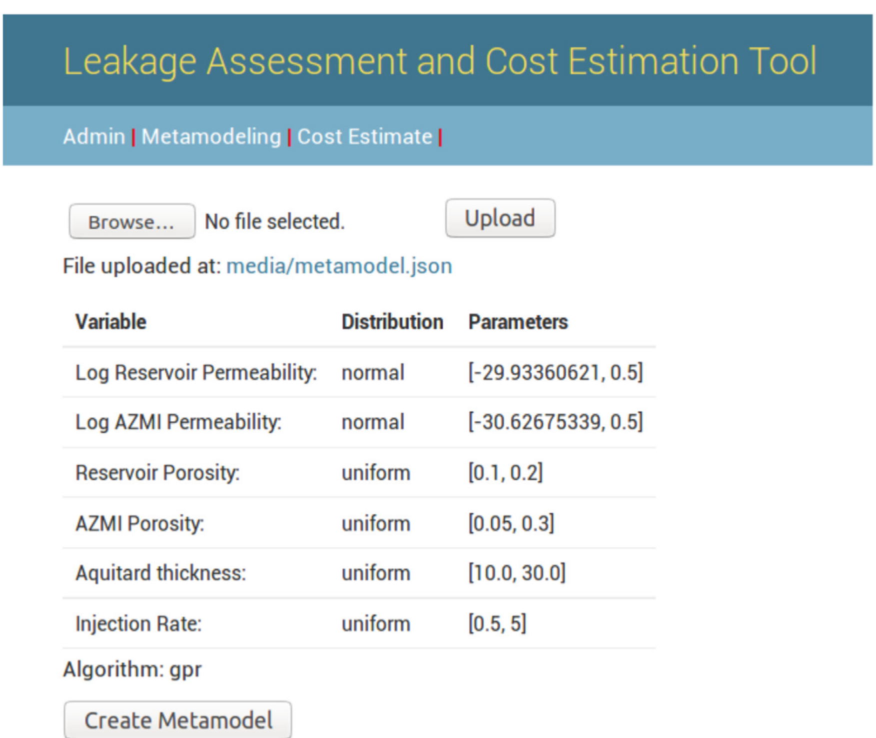

Fig. 7. User interface for uploading and displaying user-specified metamodel file, the Create Metamodel button launches the metamodel creation process.

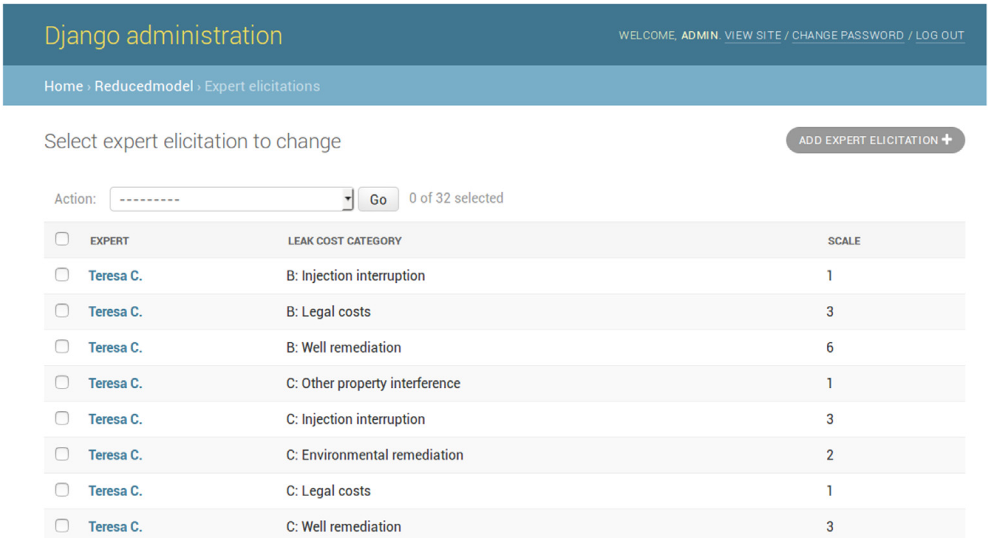

Fig. 8. User interface for soliciting expert opinions on the likelihood of each damage category, with 1 the least likely and 10 the most likely.

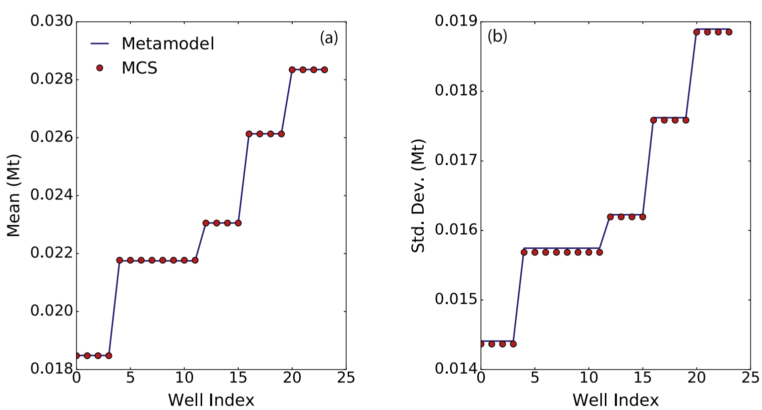

Fig. 9. Benchmark of PCE metamodel using Monte Carlo simulations: (a) mean and (b) standard deviation of total leaked CO_2 mass (in Mt). The leaky wells are sorted according to the magnitude of leakage.

given under the Distribution column, and statistical parameters are given under the Parameters column in Fig. 7. These variables are chosen because they are usually the most influential variables controlling leakage (González-Nicolás et al., 2015). In this example, the process-level CO_2 flow model is implemented as a Python module according to the algorithms described in (Nordbotten et al., 2009; Baù et al., 2015) and briefly described under Appendix A. The uncertain parameters listed in Fig. 7 are provided as input parameters to the Python module. The outputs of the model are the estimated mass of brine and CO_2 that migrate out of the injection zone.

When the Create Metamodel button (see Fig. 7) is clicked, the Django-RQ queue management module sets up asynchronous tasks to perform all required model runs, according to the sequence diagram shown in Fig. 4. Running 1000 models on TACC's Lonestar5 cluster (equipped with 1252 Cray XC40 computing nodes, with each node having two 12-core Intel Xeon processing cores) using 5 hosts (13 processes/host) generally takes less than 2 h. The results are copied back to the web server for online use.

To demonstrate cost estimation, the categories listed in Table 2 are used, the data of which are assumed to be taken from a stakeholder survey. Fig. 8 shows the user interface for entering categorical scores. The scores of all experts are aggregated and normalized to generate the weights needed for calculating the unit cost, which is shown in the rightmost column in Table 2).

3.2. Model validation

Model validation is an important step of metamodeling process (Fig. 1). Depending on the results of model validation, additional runs may be needed to improve the performance of metamodels by sampling, for instance, the underrepresented regions of parameter space. In this example, validation of the developed PCE metamodel is done by using Monte Carlo simulation, in which 1000 additional input samples are randomly generated using the joint PDF, and the outputs of PCE metamodel and process-level model are compared. Fig. 9 shows that the mean and standard deviation of total leaked CO_2 mass predicted by Monte Carlo simulation (dots) and PCE metamodel (solid line) are almost the same for all 24 leak wells.

Training and validation of the GPR metamodel is done by splitting a 1000 input sample set into two parts, 700 for training and 300 for validation. Validation of the trained GPR model on the test set shows satisfactory results, with mean Nash Sutcliff Efficiency (NSE) equal to 0.99 and mean root-mean-square-error (RMSE) equal to 0.02 kg.

3.3. Results and discussion

After a metamodel is trained and post-processed, the cost estimator can be deployed for use. Fig. 10 shows a screenshot of cost estimator implemented for the use case on hand. It serves three functional

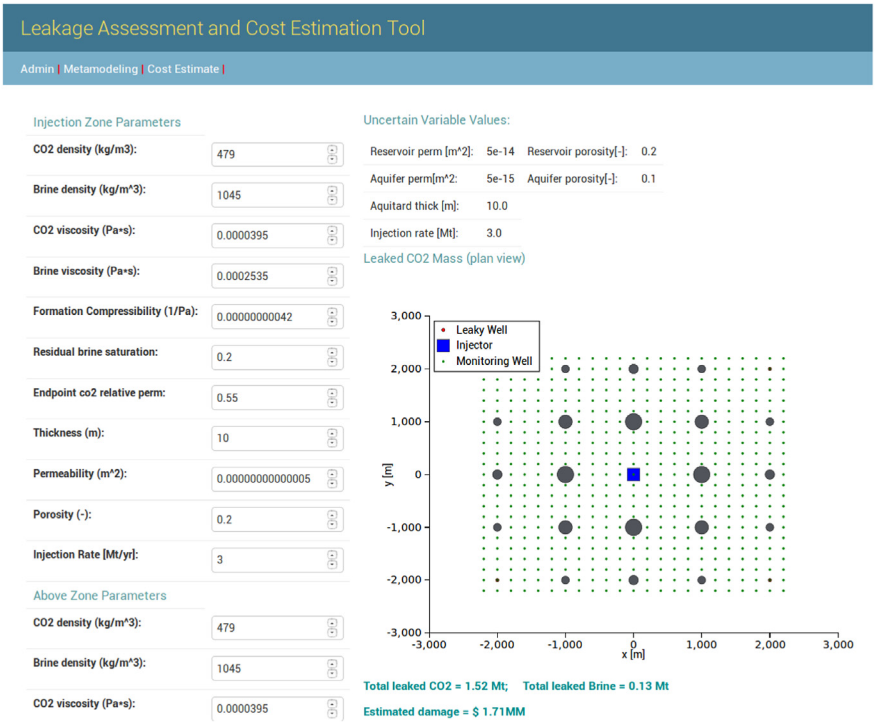

Fig. 10. Application of metamodel-supported leakage cost estimator. Left-hand panel shows user inputs and right-hand panel shows model outputs, including a plan view of leaked CO_2 mass (grey markers, their sizes are scaled according to the magnitude of leaks), and estimated total damage costs.

purposes: (1) quickly generates a spatial view of leakage predicted at all leaky well locations for any combination of uncertain parameters (in the training range), (2) estimates the damage costs, and (3) performs uncertainty quantification.

After the user clicks the Calculate button, potential leakage at all leaky well locations is plotted for a set of user inputs (see the Uncertain Variable Values part in Fig. 10; part of the web form is not shown due to its long length). The user inputs are recorded in the underlying database so they can be reloaded. For plotting, the graphics is generated dynamically using the Python package, mdlp3 (https://mpld3.github.io), which converts the native Python plot files into HTML pages. The leakage cost is estimated using the categories given in Table 2. For uncertainty quantification, PCE metamodel automatically generates the mean and variance of the leaked mass for each well, whereas the prediction variance of GPR metamodel are used to generate confidence intervals.

Because the interface between PyLACE and underlying black-box model is loosely coupled, the user may switch to any other process-level models. Modifications required for a new model may include: (1) defining a metamodel JSON file and (2) changing the Django model and view files to work with different uncertain parameter sets.

4. Summary and conclusions

Geological sequestration provides an engineering measure for reducing anthropogenic greenhouse emission. However, long-term liability costs associated with potential leakage from these geologic repositories is perceived as a barrier of entry by many interested commercial operators. An online decision support tool is necessary for multiple stakeholders to quickly assess leakage impacts, quantify uncertainty, and make business decisions collaboratively. In this work, we adopted a metamodel-mediated risk assessment approach to connect all these different tasks. A well-trained metamodel retains scientific characteristics of the original process-level model while providing a powerful tool for online decision making. The PyLACE web application developed in this work aims to automate the conversion of process-level risk assessment models into high-fidelity metamodels for online impact assessment, by utilizing high-performance computing and cloud computing infrastructures. The delegation of metamodeling to high-performance clusters significantly reduces the user burden, allowing training of the metamodels using a large input sample set. The user interface provided by the Django framework provides a direct means for the stakeholders to examine model inputs. Because of the large number of open-source tools used, package management is essential for managing version compatiability of dependencies. Finally, we emphasize that the metamodeling workflow presented here is general and can be readily applied to other process-level models by supplying user's own process-level model.

5. Software availability

PyLACE is written in Python and developed on a CentOS virtual machine. A code repository of PyLACE can be found at https://github.com/dialuser/PyLACE.

Acknowledgments

The authors are grateful to the Associate Editor (Dr. Michael Pyrcz), Dr. Ming Ye, and two other anonymous reviewers for their constructive comments. This work was supported by the U.S. Department of Energy, National Energy Technology Laboratory (NETL) under grant numbers DE-FE0012231 and DE-FE0026515.

Appendix A. ELSA Algorithm

The pressure distribution of a homogeneous brine aquifer under constant-rate CO_2 injection is (Nordbotten and Celia, 2006)

$$p(r,t,0) = p(r,t,H) - \int_0^{h(r,t)} \left[\rho_c g + \frac{q_c(r,t,z)\mu_c}{kk_{r,c}}\right] dz - \int_{h(r,t)}^{H} \left[\rho_b g + \frac{q_b(r,t,z)\mu_b}{kk_{r,b}}\right] dz \tag{A.1}$$

where ρ_α ($\alpha = b, c$ for brine and CO_2) represents fluid density; g is gravitational constant; H is aquifer thickness; q_α denotes volumetric flux; μ_α is dynamic viscosity; k is permeability; and $k_{r,\alpha}$ represents relative permeability. Assuming vertical equilibrium, the pressure buildup may be expressed as (Nordbotten et al., 2009)

$$\Delta p = \Delta p'(\rho_b - \rho_c)gH \tag{A.2}$$

and $\Delta p'$ is defined as (González-Nicolás et al., 2015)

$$\Delta p'(\chi) = \begin{cases} 0, & \chi \ge \psi \\ -\frac{1}{2\Gamma}\ln\left(\frac{\chi}{\psi}\right) + \Delta p'(\psi), & \psi > \chi \ge 2\lambda \\ \frac{1}{\Gamma} - \frac{\sqrt{\chi}}{\Gamma\sqrt{2\lambda}} + \Delta p'(2\lambda) + F(h'), & 2\lambda > \chi \ge \frac{2}{\lambda} \\ -\frac{1}{2\lambda\Gamma}\ln\left(\frac{\chi\lambda}{2}\right) + \Delta p'\left(\frac{2}{\lambda}\right), & \frac{2}{\lambda} > \chi \end{cases} \tag{A.3}$$

in which the dimensionless variables are defined as follows

$$\chi = \frac{2\pi H\phi\left(1 - S_b^{res}\right)r^2}{Qt} \tag{A.4}$$

$$\Gamma = \frac{2\pi(\rho_b - \rho_c)gkH^2}{\mu_b Q} \tag{A.5}$$

$$\psi = \frac{4.5\pi H\phi k\left(1 - S_b^{res}\right)}{\mu_b c_{eff} Q} \tag{A.6}$$

$$h' = \frac{h(\chi)}{H} = \frac{1}{\lambda - 1}\left(\sqrt{\frac{2\lambda}{\chi}} - 1\right) \tag{A.7}$$

$$F(h') = -\frac{\lambda}{\lambda - 1}\left[h' - \frac{\ln[(\lambda - 1)h' + 1]}{\lambda - 1}\right] \tag{A.8}$$

where Q is volumetric injection rate, S_b^{res} is residual brine saturation, c_{eff} is total compressibility, ϕ is porosity, and r is radial distance from the source. The flux $Q_{\alpha,j}^{(l)}$ at any leak location j and layer (l) may be expressed using Darcy's law,

$$Q_{\alpha,j}^{(l)} = -\pi r_w^2 \frac{k_w k_{r,\alpha}^{(l)}}{\mu_\alpha B^{(l)}}\left(p_j^{(l)} - p_j^{(l-1)} + \rho_\alpha g B^l + \rho_\alpha g H^{(l-1)}\right), \tag{A.9}$$

where $B^{(l)}$ is the caprock thickness, r_w is well radius, and k_w is well permeability. Using Eqs. A.3-A.9 on each observation well and each layer, a linear system of equations can be formed and solved iteratively for pressure.

Appendix B. Supplementary data

Supplementary data related to this article can be found at https://doi.org/10.1016/j.cageo.2018.01.006.

References

Arvai, J.L., 2003. Using risk communication to disclose the outcome of a participatory decision-making process: effects on the perceived acceptability of risk-policy decisions. Risk Anal. 23 (2), 281–289.
Askey, R., Wilson, J.A., 1985. Some Basic Hypergeometric Orthogonal Polynomials that Generalize Jacobi Polynomials, 319. American Mathematical Soc.
Ayyub, B.M., 2001. Elicitation of Expert Opinions for Uncertainty and Risks. CRC Press.
Baù, D., Cody, B.M., González-Nicolás, A., 2015. An iterative global pressure solution for the semi-analytical simulation of geological carbon sequestration. Comput. Geosci. 19 (4), 781–789.
Benson, S.M., Cole, D.R., 2008. CO_2 sequestration in deep sedimentary formations. Elements 4 (5), 325–331.
Bielicki, J.M., Pollak, M.F., Fitts, J.P., Peters, C.A., Wilson, E.J., 2014. Causes and financial consequences of geologic CO_2 storage reservoir leakage and interference with other subsurface resources. Int. J. Greenh. Gas Contr.l 20, 272–284.
Birkholzer, J.T., Nicot, J.P., Oldenburg, C.M., Zhou, Q., Kraemer, S., Bandilla, K., 2011. Brine flow up a well caused by pressure perturbation from geologic carbon sequestration: static and dynamic evaluations. Int. J. Greenh. Gas Contr. 5 (4), 850–861.
Birkholzer, J.T., Oldenburg, C.M., Zhou, Q., 2015. CO_2 migration and pressure evolution in deep saline aquifers. Int. J. Greenh. Gas Contr. 40, 203–220.
Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.
BP, 2016. Statistical Review of World Energy June 2016. Tech. rep., BP. https://www.bp.com/content/dam/bp/pdf/energy-economics/statistical-review-2016/bp-statistical-review-of-world-energy-2016-full-report.pdf. (Accessed 15 February 2017).
Buytaert, W., Baez, S., Bustamante, M., Dewulf, A., 2012. Web-based environmental simulation: bridging the gap between scientific modeling and decision-making. Environ. Sci. Technol. 46 (4), 1971–1976.
Carroll, S., Hao, Y., Aines, R., 2009. Geochemical detection of carbon dioxide in dilute aquifers. Geochem. Trans. 10 (1), 4.

Castronova, A.M., Goodall, J.L., Ercan, M.B., 2013. Integrated modeling within a hydrologic information system: an openmi based approach. Environ. Model. Software 39, 263–273.

Celia, M.A., Nordbotten, J.M., Court, B., Dobossy, M., Bachu, S., 2011. Field-scale application of a semi-analytical model for estimation of CO_2 and brine leakage along old wells. Int. J. Greenh. Gas Contr. 5 (2), 257–269.

Davies, L.L., Uchitel, K., Ruple, J., 2013. Understanding barriers to commercial-scale carbon capture and sequestration in the United States: an empirical assessment. Energy Pol. 59, 745–761.

Desautels, T., Krause, A., Burdick, J.W., 2014. Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. J. Mach. Learn. Res. 15 (1), 3873–3923.

Django Project, 2017. http://www.django.org, accessed March 30, 2017.

Feinberg, J., Langtangen, H.P., 2015. Chaospy: an open source tool for designing methods of uncertainty quantification. J. Comput. Sci. 11, 46–57.

Ghanem, R.G., Spanos, P.D., 2003. Stochastic Finite Elements: a Spectral Approach. Courier Corporation.

Gladwell, M., 2006. The Tipping Point: How Little Things Can Make a Big Difference. Little, Brown.

González-Nicolás, A., Baù, D., Cody, B.M., Alzraiee, A., 2015. Stochastic and global sensitivity analyses of uncertain parameters affecting the safety of geological carbon storage in saline aquifers of the Michigan basin. Int. J. Greenh. Gas Contr. 37, 99–114.

Goodall, J.L., Robinson, B.F., Castronova, A.M., 2011. Modeling water resource systems using a service-oriented computing paradigm. Environ. Model. Software 26 (5), 573–582.

GPy, 2012. GPy: A Gaussian Process Framework in Python. Sheffield Machine Learning Group. http://github.com/SheffieldML/GPy.

Harbert, W., Daley, T.M., Bromhal, G., Sullivan, C., Huang, L., 2016. Progress in monitoring strategies for risk reduction in geologic CO_2 storage. Int. J. Greenh. Gas Contr. 51, 260–275.

Haszeldine, R.S., 2009. Carbon capture and storage: how green can black be? Science 325 (5948), 1647–1652.

Islam, A.W., Sun, A.Y., 2016. Corrosion model of co 2 injection based on non-isothermal wellbore hydraulics. International Journal of Greenhouse Gas Control 54, 219–227.

Jenkins, C., Chadwick, A., Hovorka, S.D., 2015. The state of the art in monitoring and verification—ten years on. Int. J. Greenh. Gas Contr. 40, 312–349.

Jordan, P.D., Benson, S.M., 2009. Well blowout rates and consequences in California oil and gas district 4 from 1991 to 2005: implications for geological storage of carbon dioxide. Environ. Geol. 57 (5), 1103–1123.

Krause, A., Guestrin, C., 2007. Nonmyopic active learning of Gaussian processes: an exploration-exploitation approach. In: Proceedings of the 24th International Conference on Machine Learning. ACM, pp. 449–456.

Krueger, T., Page, T., Hubacek, K., Smith, L., Hiscock, K., 2012. The role of expert opinion in environmental modelling. Environ. Model. Software 36, 4–18.

Laniak, G.F., Olchin, G., Goodall, J., Voinov, A., Hill, M., Glynn, P., Whelan, G., Geller, G., Quinn, N., Blind, M., et al., 2013. Integrated environmental modeling: a vision and roadmap for the future. Environ. Model. Software 39, 3–23.

Li, H., Zhang, D., 2007. Probabilistic collocation method for flow in porous media: comparisons with other stochastic methods. Water Resour. Res. 43 (9).

Maier, H.R., Kapelan, Z., Kasprzyk, J., Kollat, J., Matott, L., Cunha, M., Dandy, G.C., Gibbs, M.S., Keedwell, E., Marchi, A., et al., 2014. Evolutionary algorithms and other metaheuristics in water resources: current status, research challenges and future directions. Environ. Model. Software 62, 271–299.

Marrel, A., Iooss, B., Van Dorpe, F., Volkova, E., 2008. An efficient methodology for modeling complex computer codes with Gaussian processes. Comput. Stat. Data Anal. 52 (10), 4731–4744.

Matthies, M., Giupponi, C., Ostendorf, B., 2007. Environmental decision support systems: current issues, methods and tools. Environ. Model. Software 22 (2), 123–127.

Navarre-Sitchler, A.K., Maxwell, R.M., Siirila, E.R., Hammond, G.E., Lichtner, P.C., 2013. Elucidating geochemical response of shallow heterogeneous aquifers to co 2 leakage using high-performance computing: implications for monitoring of co_2 sequestration. Adv. Water Resour. 53, 45–55.

Nordbotten, J., Kavetski, D., Celia, M., Bachu, S., 2009. A semi-analytical model estimating leakage associated with co_2 storage in large-scale multi-layered geological systems with multiple leaky wells. Environ. Sci. Technol. 43 (3), 743–749.

Nordbotten, J.M., Celia, M.A., 2006. Similarity solutions for fluid injection into confined aquifers. J. Fluid Mech. 561, 307–327.

NRC, 1996. Understanding Risk: Informing Decisions in a Democratic Society. National Academy Press, Washington, DC.

Oldenburg, C.M., Bryant, S.L., Nicot, J.-P., 2009. Certification framework based on effective trapping for geologic carbon sequestration. Int. J. Greenh. Gas Contr. 3 (4), 444–457.

Pawar, R.J., Bromhal, G.S., Carey, J.W., Foxall, W., Korre, A., Ringrose, P.S., Tucker, O., Watson, M.N., White, J.A., 2015. Recent advances in risk assessment and risk management of geologic CO_2 storage. Int. J. Greenh. Gas Contr. 40, 292–311.

Pollak, M., McCoy, S.T., 2011. Monitoring for greenhouse gas accounting at geologic sequestration sites: technical and policy considerations. Energy Procedia 4, 5917–5924.

Rajib, M.A., Merwade, V., Kim, I.L., Zhao, L., Song, C., Zhe, S., 2016. Swatshare–a web platform for collaborative research and education through online sharing, simulation and visualization of swat models. Environ. Model. Software 75, 498–512.

Rasmussen, C.E., 2006. Gaussian Processes for Machine Learning.

Razavi, S., Tolson, B.A., Burn, D.H., 2012. Review of surrogate modeling in water resources. Water Resour. Res. 48 (7).

Rehr, A.P., Small, M.J., Bradley, P., Fisher, W.S., Vega, A., Black, K., Stockton, T., 2012. A decision support framework for science-based, multi-stakeholder deliberation: a coral reef example. Environ. Manag. 50 (6), 1204–1218.

Rosenblatt, M., 1952. Remarks on a multivariate transformation. Ann. Math. Stat. 23 (3), 470–472.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., Tarantola, S., 2008. Global Sensitivity Analysis: the Primer. John Wiley & Sons.

Stauffer, P.H., Viswanathan, H., Pawar, R.J., Klasky, M.L., Guthrie, G.D., 2006. CO_2-pens: a CO_2 sequestration systems model supporting risk-based decisions. In: Proceedings of the 16th International Conference on Computational Methods in Water Resources, pp. 19–22.

Sudret, B., 2008. Global sensitivity analysis using polynomial chaos expansions. Reliab. Eng. Syst. Saf. 93 (7), 964–979.

Sun, A.Y., 2013. Enabling collaborative decision-making in watershed management using cloud-computing services. Environ. Model. Software 41, 93–97.

Sun, A.Y., Miranda, R.M., Xu, X., 2015. Development of multi-metamodels to support surface water quality management and decision making. Environ. Earth Sci. 73 (1), 423–434.

Sun, A.Y., Nicot, J.-P., Zhang, X., 2013a. Optimal design of pressure-based, leakage detection monitoring networks for geologic carbon sequestration repositories. Int. Journal of Greenhouse Gas Control 19, 251–261.

Sun, A.Y., Wang, D., Xu, X., 2014. Monthly streamflow forecasting using Gaussian process regression. J. Hydrol. 511, 72–81.

Sun, A.Y., Zeidouni, M., Nicot, J.-P., Lu, Z., Zhang, D., 2013b. Assessing leakage detectability at geologic CO_2 sequestration sites using the probabilistic collocation method. Adv. Water Resour. 56, 49–60.

Sun, N.-Z., Sun, A.Y., 2015. Model Calibration and Parameter Estimation: for Environmental and Water Resource Systems. Springer.

Swain, N.R., Christensen, S.D., Snow, A.D., Dolder, H., Espinoza-Dávalos, G., Goharian, E., Jones, N.L., Nelson, E.J., Ames, D.P., Burian, S.J., 2016. A new open source platform for lowering the barrier for environmental web app development. Environ. Model. Software 85, 11–26.

Trabucchi, C., Donlan, M., Wade, S., 2010. A multi-disciplinary framework to monetize financial consequences arising from ccs projects and motivate effective financial responsibility. Int. J. Greenh. Gas Contr. 4 (2), 388–395.

Uusitalo, L., Lehikoinen, A., Helle, I., Myrberg, K., 2015. An overview of methods to evaluate uncertainty of deterministic models in decision support. Environ. Model. Software 63, 24–31.

Villa-Vialaneix, N., Follador, M., Ratto, M., Leip, A., 2012. A comparison of eight metamodeling techniques for the simulation of N_2O fluxes and n leaching from corn crops. Environ. Model. Software 34, 51–66.

Viswanathan, H.S., Pawar, R.J., Stauffer, P.H., Kaszuba, J.P., Carey, J.W., Olsen, S.C., Keating, G.N., Kavetski, D., Guthrie, G.D., 2008. Development of a hybrid process and system model for the assessment of wellbore leakage at a geologic co_2 sequestration site. Environ. Sci. Technol. 42 (19), 7280–7286.

Vitolo, C., Elkhatib, Y., Reusser, D., Macleod, C.J., Buytaert, W., 2015. Web technologies for environmental big data. Environ. Model. Software 63, 185–198.

Wang, G.G., Shan, S., 2007. Review of metamodeling techniques in support of engineering design optimization. J. Mech. Des. 129 (4), 370–380.

Wiener, N., 1938. The homogeneous chaos. Am. J. Math. 60 (4), 897–936.

Wilson, E.J., Friedmann, S.J., Pollak, M.F., 2007. Research for deployment: incorporating risk, regulation, and liability for carbon capture and sequestration. Environ. Sci. Technol. 41 (17), 5945–5952 pMID: 17937265.

Wilson, E.J., Johnson, T.L., Keith, D.W., 2003. Regulating the ultimate sink: managing the risks of geologic CO_2 storage. Environ. Sci. Technol. 37 (16), 3476–3483 pMID: 12953855.

Wilson, L.A., Fonner, J.M., 2014. Launcher: a shell-based framework for rapid development of parallel parametric studies. In: Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment. ACM, p. 40.

Xiu, D., Karniadakis, G.E., 2002. The wiener–askey polynomial chaos for stochastic differential equations. SIAM J. Sci. Comput. 24 (2), 619–644.

Zeng, L., Shi, L., Zhang, D., Wu, L., 2012. A sparse grid based bayesian method for contaminant source identification. Adv. Water Resour. 37, 1–9.

Zhang, G., Lu, D., Ye, M., Gunzburger, M., Webster, C., 2013. An adaptive sparse-grid high-order stochastic collocation method for bayesian inference in groundwater reactive transport modeling. Water Resour. Res. 49 (10), 6871–6892.

Zhang, J., Li, W., Zeng, L., Wu, L., 2016. An adaptive Gaussian process-based method for efficient bayesian experimental design in groundwater contaminant source identification problems. Water Resour. Res. 52 (8), 5971–5984.

Zhou, Q., Birkholzer, J.T., Tsang, C.-F., Rutqvist, J., 2008. A method for quick assessment of CO_2 storage capacity in closed and semi-closed saline formations. Int. J. Greenh. Gas Contr. 2 (4), 626–639.