Computers and Geotechnics 90 (2017) 164–175

Contents lists available at ScienceDirect

Computers and Geotechnics

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

Research Paper

Strategy for consistent model parameter calibration for soft soils using multi-objective optimisation

J.-P. Gras [a], N. Sivasithamparam [b], M. Karstunen [a], J. Dijkstra [a,*]

[a] Chalmers University of Technology, Department of Civil and Environmental Engineering, SE-412 96 Gothenburg, Sweden
[b] Norwegian Geotechnical Institute, P.O. Box 3930, Ullevål Stadion, N-0806 Oslo, Norway

ARTICLE INFO

Article history:
Received 13 November 2016
Received in revised form 8 May 2017
Accepted 7 June 2017
Available online 23 June 2017

Keywords:
Soft soil
CREEP-SCLAY1S
Sensitivity analysis
Multi-objective optimisation

ABSTRACT

Constitutive models for soft soils require a large number of parameters to model the complex material response. One set of parameters should capture the soil response for various laboratory & in situ stress paths. This requires a new method to calibrate a consistent set of model parameters using test data from different load paths of classic geomechanical tests. The feasibility of the proposed method is demonstrated with the recently developed CREEP-SCLAY1S model. After a sensitivity analysis that highlights the model parameters for optimisation, an optimisation process for two different configurations is designed. The latter is successfully verified against artificially generated laboratory data.

1. Introduction

Soft clays are very complex materials. Advanced models taking into account anisotropy, time/rate dependency, bonding/destructuration and non-linear elasticity are required to correctly describe and predict their mechanical behaviour. An unfortunate side effect is that these advanced models require a large number of parameters. Generally, in such models, most of these parameters are directly assessed from a single type of laboratory test, but some cannot be measured experimentally. It is then necessary to estimate these parameters through indirect methods such as calibration [1].

Automatic methods for model calibration are essential in order to identify model parameters independently of the user. Often these methods are based on the inverse problem theory by [2]. It consists of finding the best set of parameters which minimises the distance between experimental results and model results. Numerous methods for inverse analysis for geotechnical problems have been carried out with gradient methods [3–6], neural network based techniques [7,8], genetic algorithms [9,10] and particle swarm optimisation [11–13].

Generally, the parameter identification techniques based on inverse analysis remain unsatisfactory because they do not sufficiently take into account the non-uniqueness of the inverse analysis problems. This difficulty can be overcome by determining a set of satisfactory solutions (multi-objective optimisation process) or the use of several test types with different load paths [14]. The difficulty to obtain the uniqueness of the solution is increasing with the number of parameters to optimise. Practically, it is not feasible to obtain all parameter values for complex models with many parameters. There is then a need to choose the proper parameters for optimisation. Principally, the optimisation parameters are only the parameters which are not measurable experimentally. However, even the parameters that are directly derived from experimental data have some measurement uncertainties which will affect the final output.

In this article, the uniqueness of the solution is studied using artificial data. The optimisation method uses a genetic algorithm combined with selected relevant test paths to obtain reliable model parameter solutions. In order to choose the parameters for optimisation, a sensitivity analysis was performed for different test paths with a different set of model parameters and initial conditions. From the sensitivity analysis, the most important parameters, which need to be optimised, are identified.

2. Software framework

An overview of the principle work flow of the software tool is shown in Fig. 1. Three important stages are highlighted in grey. Two of those, i.e. the sensitivity analysis and optimisation of

* Corresponding author.
E-mail addresses: jean-philippe.gras@chalmers.se (J.-P. Gras), nallathamby.siva@ngi.no (N. Sivasithamparam), minna.karstunen@chalmers.se (M. Karstunen), jelke.dijkstra@chalmers.se (J. Dijkstra).

http://dx.doi.org/10.1016/j.compgeo.2017.06.006
0266-352X/

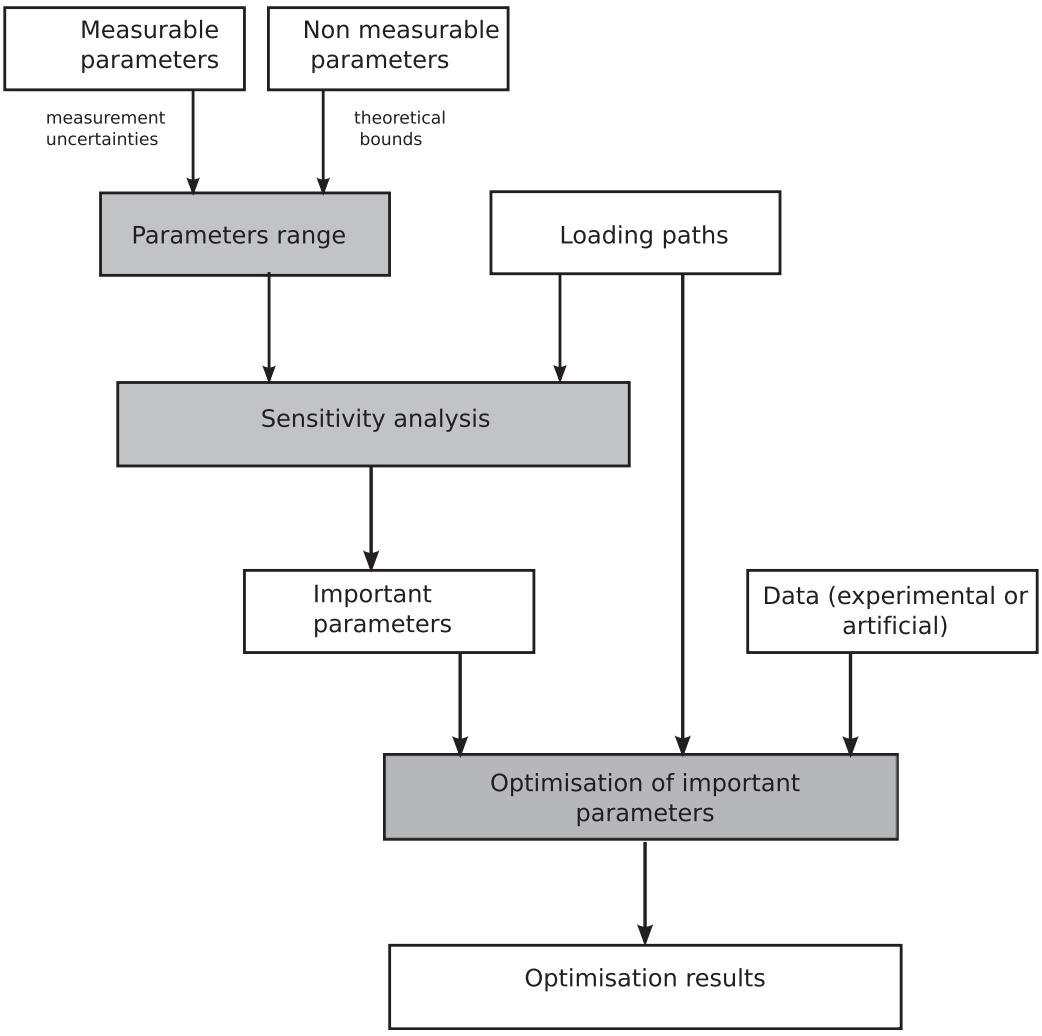

Fig. 1. Flow chart of the optimisation method.

parameters, are implemented as two modules in the tool. In contrast the third, the selection of the parameter bounds, is limited by the model formulation and potential measurement errors and requires further user input as is elaborated in [15].

The software tool is in fact a graphical user interface (GUI) wrapped around an incremental driver, developed by [16,17], using the Python PyQt library. The incremental driver enables probing the constitutive model on a single integration point level for arbitrary stress and strain paths and is compatible with the syntax of the user-defined material subroutine of ABAQUS, i.e. UMAT. The implementation is universal where the Python code calls the incremental driver, which is a standalone executable, see Fig. 2. In principle any constitutive model implemented as a UMAT library can be linked to the incremental driver, and hence the Python wrapper. Furthermore, this enables straightforward expansion towards parallelisation with multiple threads, each one calling an incremental driver. In the current implementation the number of loading paths that can be evaluated is limited by the available test data rather than computational limitations.

The Python GUI allows for easy access to additional libraries for sensitivity analyses and model optimisation. The sensitivity analysis tool to study the importance and sensitivity of parameters of the constitutive model in the strain driver is implemented using the Sensitivity Analysis Library, the SALib package [18]. The optimisation tool, to optimise model parameters against experimental data, has been implemented via the Distributed Evolutionary Algorithms in Python, DEAP, library [19]. Both tools may be used for one loading path or multiple loading paths.

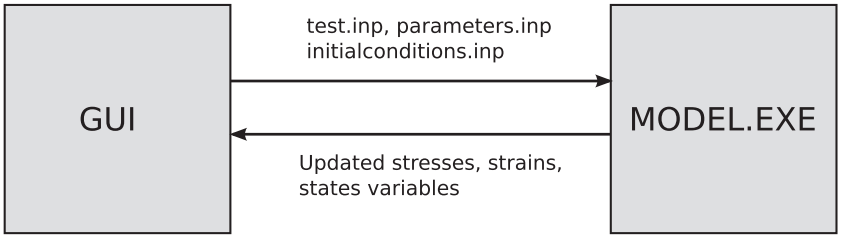

Fig. 2. Communication between the Graphical User Interface (GUI) wrapper script and the incremental driver.