Pontifícia Universidade Católica
do Rio de Janeiro

Felipe Tajá Costa Pinto

Processo de Calibração dos Microparâmetros em Método de Elementos Discretos via Generalized Simulated Annealing

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Engenharia de Civil e Ambiental da PUC-Rio.

Orientador: Profa. Raquel Quadros Velloso

Rio de Janeiro
Julho de 2020

PUC-Rio - Certificação Digital Nº 1712747/CA

Pontifícia Universidade Católica
do Rio de Janeiro

Felipe Tajá Costa Pinto

Processo de Calibração dos Microparâmetros em Método de Elementos Discretos via Generalized Simulated Annealing

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Engenharia de Civil e Ambiental da PUC-Rio. Aprovada pela Comissão Examinadora.

Profa. Raquel Quadros Velloso
Orientador
Departamento de Engenharia de Civil e Ambiental – PUC-Rio

Prof. Eurípedes do Amaral Vargas Jr.
Departamento de Engenharia Civil e Ambiental – PUC-Rio

Flávia de Oliveira Lima Falcão
Petróleo Brasileiro SA – Petrobras

Rio de Janeiro, 20 de Julho de 2020

PUC-Rio - Certificação Digital Nº 1712747/CA

Todos os direitos reservados. É proibida a reprodução total ou parcial do trabalho sem autorização da universidade, do autor e do orientador.

Felipe Tajá Costa Pinto

Graduado e Mestre em Física pela Universidade Federal da Bahia na área de Matéria Condensada. Na Petrobras trabalhou como Geofísico Fiscal em equipe sísmica na Amazônia e atualmente trabalha com modelagens de Geopressões e Geomecânica para construção de poços de Petróleo.

PUC-Rio - Certificação Digital Nº 1712747/CA

Ficha Catalográfica

Pinto, Felipe Tajá Costa

Processo de Calibração dos Microparâmetros em Método de Elementos Discretos via Generalized Simulated Annealing / Felipe Tajá Costa Pinto; orientador: Raquel Quadros Velloso. – Rio de janeiro: PUC-Rio, Departamento de Engenharia de Civil e Ambiental , 2020.

v., 80 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia de Civil e Ambiental .

Inclui bibliografia

1. Engenharia Civil – Teses. 2. Geotecnia – Teses. 3. Método de Elementos Discretos;. 4. Microparâmetros;. 5. Calibração;. 6. Otimização;. 7. Generalized Simulated Annealing.. I. Velloso, Raquel Quadros. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia de Civil e Ambiental . III. Título.

CDD: 624

PUC-Rio - Certificação Digital Nº 1712747/CA

Dedico este texto a meu pequeno Bruno.
Que você venha gostar de Ciências tanto quanto eu.

Agradecimentos

Nenhum trabalho científico poderia ser realizado sem qualquer tipo de ajuda. Gostaria de agradecer nominalmente às pessoas que contribuíram para que esta dissertação pudesse ser concluída.

Em primeiro lugar, à minha esposa, companheira e amiga Vanessa por estar comigo todos estes anos e por, durante o curso, ter segurando a barra quando necessário;

Ao meu pequeno filho Bruno por ter nascido durante o período do mestrado, fazendo com que eu pudesse relaxar um pouco e curti-lo mais nesta fase inicial da sua vida;

Aos meu pais Elisabete e Reinério (*in memoriam*) e minha irmã Tiaia que me permitiram ser quem eu sou hoje;

À Profa. Dra. Raquel Velloso por ter acreditado na minha proposta de trabalho, por todas as dicas e contatos fornecidos durante esta jornada;

Aos meus antigos chefes Sebastião Pereira e Marcos Fetter por terem me incentivado e permitido a realização deste curso. Foi excelente ter líderes tão agradáveis, companheiros e próximos de suas equipes;

Aos meu amigos Angelo Corrêa e Gilmar Alves que, via nosso grupo Filosofia Natural, nos permitiram ter discussões, sugestões e elucidado várias das minhas dúvidas além de ter revisado o texto mesmo sem entender quase nada do assunto. Viver agora no futuro é tão maravilhoso;

Julio Rueda e Marcelo de Simone pela grande ajuda na correção e dicas no meu *script* do Yade;

Ao colega Antônio de Pádua pela primeira aula que tive na vida sobre Python;

Aos colegas da Geomecânica: Marcos “Ravenga” Domingues, Tiago Sobrinho, Thiago Lopes, Richard Guimarães, Egydio Chianello e aos colegas do quase extinto feudo Garcia: Maurício “Wally” Garcia, Paulo Kerber “Garcia”, Vinícius Carneiro “Garcia”, Fabrício Kacinskas “Garcia” pelas ajudas na seleção de alguns dados, nas correções texto em inglês e na estatística descritiva.

Principalmente agradecer a Marquinhos Santini que me ativou a memória para o uso da integral para comparar duas funções além das excelente sugestões no texto;

Um agradecimento especial aos dois colegas de trabalho que eu mais admiro pela gentileza, grandeza do conhecimento, capacidade de transmissão de informações complexas de uma maneira simples e as inúmeras sugestões que melhoraram substancialmente o meu texto: Guilherme Bispo e Julio Garcia;

Aos colegas de empresa e de estudos: Leandro Guedes e chefe Tatiana Oliveira pelas horas que passamos tentando entender as matérias;

PUC-Rio - Certificação Digital Nº 1712747/CA

Ao meu primo, compadre e amigo desde... sempre: Caio “Caiteiro” Costa por, mais uma vez, me ajudar nas figuras e ter paciência em me escutar reclamar de tudo;

Ao meu antigo orientador Prof. Dr. Caio Castilho por ter me ensinado a fazer ciência;

À Petrobras por esta grande oportunidade de aperfeiçoamento.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

PUC-Rio - Certificação Digital Nº 1712747/CA

Resumo

Pinto, Felipe Tajá Costa; Velloso, Raquel Quadros. **Processo de Calibração dos Microparâmetros em Método de Elementos Discretos via Generalized Simulated Annealing**. Rio de Janeiro, 2020. 80 p. Dissertação de Mestrado – Departamento de Engenharia de Civil e Ambiental , Pontifícia Universidade Católica do Rio de Janeiro.

O Método dos Elementos Discretos (*Discrete Element Method* - DEM) é uma técnica numérico computacional capaz de simular o comportamento macroscópico de um material via solução das equações do movimento de seus constituintes. Para uma correta predição deste comportamento são informados, como dados de entrada, as características mecânicas dos elementos: os chamados microparâmetros. Contudo, não existe uma receita que determine estes microparâmetros baseados somente nas respostas macroscópicas do material simulado, necessitando de um passo adicional conhecido como Calibração. Tentativa e erro, um método ineficiente por conta de seu fator de escala desfavorável, é o mais comumente utilizado nesta etapa. Este trabalho propõe uma nova abordagem utilizando-se do método de otimização global *Generalized Simulated Annealing*, minimizando-se a área quadrática normalizada entre as curvas experimentais e calculadas de tensão-deformação axial e deformações volumétrica-axial simultaneamente. Foram efetuadas comparações via ensaio triaxial para dados sintéticos e reais cujos resultados demonstram o aproveitamento e aplicabilidade da técnica proposta.

PUC-Rio - Certificação Digital Nº 1712747/CA

Palavras-chave

Método de Elementos Discretos; Microparâmetros; Calibração; Otimização; Generalized Simulated Annealing.

Abstract

Pinto, Felipe Tajá Costa; Velloso, Raquel Quadros (Advisor). **Microparameters Calibration Process in DEM via Generalized Simulated Annealing**. Rio de Janeiro, 2020. 80 p. Dissertação de mestrado – Departamento de Engenharia de Civil e Ambiental , Pontifícia Universidade Católica do Rio de Janeiro.

The Discrete Element Method (DEM) is a numerical computational technique that simulates the macroscopic material behaviour by solving the equations of motion of its constituents. For a correct prediction of this behaviour, are set as input data the mechanical characteristics of the elements, the so-called microparameters. However, there is no recipe for determining these microparameters based solely on the macroscopic responses of the simulated material. It is required an additional step known as Calibration. The method widely used in this calibration is trial and error, although is an inefficient method due its unfavorable scale factor. This work proposes a new approach using the Generalized Simulated Annealing global optimization method, minimizing the normalized quadratic area between the experimental and calculated curves of the axial stress-strain and volumetric-axial deformations curves simultaneously. Comparison is done using triaxial tests for both synthetic and real data whose results demonstrate the usefulness and applicability of the proposed approach.

PUC-Rio - Certificação Digital Nº 1712747/CA

Keywords

Discrete Element Method; Microparameters; Calibration; Optimization; Generalized Simulated Annealing.

Sumário

1 Introdução **1**

2 Revisão Bibliográfica **3**
2.1 DEM 3
2.1.1 Modelo de Contato 6
2.2 Uso de Otimização em DEM 9
2.3 Generalized Simulated Annealing 11
2.3.1 Função Distribuição ($g_{q_V}(\delta\vec{r})$) 15
2.3.2 Esquema de Resfriamento ($T_{q_V}^V(t)$) 16
2.3.3 Probabilidade de Aceitação ($P_{q_A}(\vec{r}_t \rightarrow \vec{r}_{t+1})$) 17
2.4 Função Objetivo 18

3 Metodologia **20**
3.1 Yade 20
3.2 DEM 20
3.2.1 Dimensões das Amostras 21
3.2.2 Densidade (ρ) 24
3.2.3 *Interaction Range* (γ_{int}) 24
3.2.4 Velocidade de Carregamento (v_l) 25
3.2.5 O Fator de Amortecimento Numérico (λ_d) 25
3.2.6 Microparâmetros 26
3.2.7 Análise de Tempo de Processamento 26
3.2.8 Indeterminismo Devido ao Multiprocessamento 29
3.3 *Generalized Simulated Annealing* 30
3.3.1 Par (q_V; q_A) 30
3.3.2 Temperatura Inicial (T_0) 31
3.3.3 Critérios de Parada 31
3.3.3.1 Convegência da Função Objetivo 32
3.3.3.2 Número Máximo de Cadeias de Markov 32
3.4 Aspectos Computacionais 33

4 Resultados e Discussão **34**
4.1 Dados Sintéticos 34
4.2 Dados Reais 36

5 Conclusões **39**

6 Sugestões Para Trabalhos Futuros **40**

7 Referências **41**

A Demonstração do Mínimo para o Funcional R_2 **47**

B Código Fonte do Ensaio Triaxial **48**

PUC-Rio - Certificação Digital Nº 1712747/CA

PUC-Rio - Certificação Digital Nº 1712747/CA

Lista de Figuras

Figura 2.1 As diversas geometrias utilizadas em simulações via DEM. 4
2.1(a) Geometria alongada. 4
2.1(b) Aglomerado de esferas ou *clumps*. 4
2.1(c) Geometrias elipsoidal, cúbica e cilíndrica. 4
2.1(d) Geometria tridimensional superquadrática. Os parâmetros ϵ_1 e ϵ_2 são os definidores de forma dos elementos. 4
Figura 2.2 O esquema geral de cálculo em DEM. 5
Figura 2.3 Representação geométrica do parâmetro γ_{int}. 7
Figura 2.4 Representação da força normal em dois elementos discretos. 7
Figura 2.5 Representação da força cisalhante entre dois elementos discretos. 8
Figura 2.6 Modelo de Mohr-Coulomb modificado conforme modelo de contato JCF. 8
Figura 2.7 Fluxograma do *Simulated Annealing*. O passo em destaque é onde se insere a função objetivo, conforme o problema proposto. A série de cálculos efetuados com uma mesma temperatura é chamada de cadeia de Markov. 14
Figura 2.8 Gráfico da função distribuição de Tsallis para diversos valores de q_V normalizados com respeito à área sob as curvas. Este exemplo foi gerado com $D = 1$ e $T = 1$. Observa-se que, à medida em que os valores de q_V aumentam, a função torna-se cada vez mais uma distribuição de cauda longa, ou seja, uma maior probabilidade de haver sorteios mais distantes do centro da distribuição. 15
Figura 2.9 Representação unidimensional das distribuições uniforme, gaussiana e de Cauchy-Lorentz. 16
Figura 2.10 Gráficos do esquema de resfriamento, ou função temperatura generalizada, em função do tempo (ou número de cadeias de Markov), para vários valores de q_V. Este gráfico foi gerado com $T_0 = 100$. À medida em que se aumenta o valor do parâmetro q_V, o resfriamento tende a ser mais severo. 17
Figura 2.11 Gráfico da probabilidade de aceitação em função da diferença da função objetivo. Para valores de q_A menores há uma tendência a aceitar menos passos de subida, aumentando a eficiência. 18
Figura 2.12 O funcional R_2 representa a área quadrática normalizada entre as curvas experimental e calculada. 19
2.12(a) Gráficos de tensão *vs* deformação axial para dois ensaios uniaxial (experimental e calculado). 19
2.12(b) Gráficos da deformação volumétrica *vs* deformação axial para o mesmo ensaio. 19

PUC-Rio - Certificação Digital Nº 1712747/CA

Figura 3.1 Amostra utilizada nos ensaios triaxiais. O centro e raio das esferas foram geradas aleatoriamente. As cores vermelho e verde das extremidades indicam os locais onde serão aplicadas as tensões σ_1. 22
3.1(a)Vista da face paralela ao plano xy. 22
3.1(b)Vista da face paralela ao plano yz. 22
3.1(c)Vista da face paralela ao plano zx. 22
Figura 3.2 Cruvas do ensaio triaxial do arenito Fontainebleau com tensões confinantes de 1 MPa, 5 MPa e 10 MPa, calculado com os microparâmetros da Tabela 3.2. Estas curvas serão chamadas de experimentais. 23
3.2(a)Gráficos $\sigma_d \times \epsilon_a$. 23
3.2(b)Gráficos $\epsilon_V \times \epsilon_a$. 23
3.2(c)Gráfico $\sigma_1 \times \sigma_3$ de pico. 23
Figura 3.3 Gráfico de R_2 em função de λ_d. 25
Figura 3.4 Análise de sensibilidade dos microparâmetros mais relevantes para a função objetivo: E_m, ϕ_b e t. A escala do eixo R_2 varia de -0,1 a 0,7. 28
3.4(a)Gráfico $R_2 \times E_m$. 28
3.4(b)Gráfico $R_2 \times \phi_b$. 28
3.4(c)Gráfico $R_2 \times t$. 28
Figura 3.5 Análise de sensibilidade dos microparâmetros menos relevantes para a função objetivo: c e k_s/k_n. A escala do eixo R_2 varia de -0,0001 a 0,0012. 29
3.5(a)Gráfico $R_2 \times c$. 29
3.5(b)Gráfico $R_2 \times k_s/k_n$. 29
Figura 3.6 Tempo de processamento em função do número de processadores para a simulação do ensaio triaxial com tensão confinantes variadas. Estes testes foi realizados numa máquina equipada com processador Intel Xeon CPU E5-2687W 0 @ 3.10 GHz. 30
3.6(a)$\sigma_3 = 1$ MPa. 30
3.6(b)$\sigma_3 = 5$ MPa. 30
3.6(c)$\sigma_3 = 10$ MPa. 30
Figura 3.7 Histograma do sorteio aleatório para várias temperaturas distintas. Pode-se perceber que a amplitude da distribuição aumenta à medida que se aumenta a temperatura. 32
3.7(a)$T_0 = 0{,}1$ 32
3.7(b)$T_0 = 1{,}0$ 32
3.7(c)$T_0 = 10{,}0$ 32
3.7(d)$T_0 = 20{,}0$ 32

Figura 4.1 Evolução do valor mínimo da função objetivo nas três otimizações realizadas. 35
Figura 4.2 Comparação das curvas $\sigma_d \times \epsilon_a$ experimentais e otimizadas para as tensões confinantes de 1 MPa, 5 MPa e 10 MPa. 35
4.2(a)$\sigma_3 = 1$ MPa. 35
4.2(b)$\sigma_3 = 5$ MPa. 35
4.2(c)$\sigma_3 = 10$ MPa. 35

PUC-Rio - Certificação Digital Nº 1712747/CA

Figura 4.3 Comparação das curvas $\epsilon_V \times \epsilon_a$ experimentais e otimizadas para as tensões confinantes de 1 MPa, 5 MPa e 10 MPa. 36
4.3(a)$\sigma_3 = 1$ MPa. 36
4.3(b)$\sigma_3 = 5$ MPa. 36
4.3(c)$\sigma_3 = 10$ MPa. 36
Figura 4.4 Evolução do valor mínimo da função objetivo nas três otimizações com dados reais do Travertino Romano. Cada otimização finalizou em $R_2 = 0{,}2577$ (preto), $R_2 = 0{,}3377$ (verde) e $R_2 = 0{,}6153$ (vermelho), resultando numa média de $R_2 = 0{,}4 \pm 0{,}2$. 37
Figura 4.5 Comparação das curvas experimentais e calculadas após a otimização via GSA. Foram desenhadas as curvas otimizadas com o menor valor da função objetivo ($R_2 = 0{,}2577$). 38
4.5(a)Gráficos de tensão-deformação axial experimental (preto) e otimizada (vermelho). 38
4.5(b)Gráficos de deformações volumétrica-axial experimental (preto) e otimizada (vermelho). 38

PUC-Rio - Certificação Digital Nº 1712747/CA

Lista de Tabelas

Tabela 3.1 Sugestões de preparação de amostra conforme a ISRM e a ASTM para ensaios de compressão. 21
Tabela 3.2 Microparâmetros para o arenito Fontainebleau. 22
Tabela 3.3 Macroparâmetros obtidos dos ensaios do arenito Fontainebleau com os dados da Tabela 3.2. 22
Tabela 3.4 Valores de R_2 em função da densidade da amostra. A função objetivo R_2 foi calculada comparando-se as curvas calculadas com as experimentais cuja a densidade foi de 2700 kg/m^3. 24
Tabela 3.5 Compilação do resultado da análise de sensibilidade de R_2 usando os microparâmetros como dados de entrada nas simulações. Exceto onde indicado, os microparâmetros foram mantidos fixos conforme os valores da Tabela 3.2. 27

Tabela 4.1 Microparâmetros otimizados via GSA. 34
Tabela 4.2 Respostas macroscópicas do material simulado após a otimização. 34
Tabela 4.3 Limites de cada variável na otimização dos dados do Travertino Romano. 36
Tabela 4.4 Variáveis otimizadas para o Travertino Romano via GSA. 37
Tabela 4.5 Respostas macroscópica do material simulado após a otimização. Os valores foram obtidos a partir das curvas com o $R_2 = 0{,}2577$. 38

PUC-Rio - Certificação Digital Nº 1712747/CA

Lista de Símbolos

Abreviações

DEM	*Discrete Element Method*
JCF	Modelo de contato *Jointed Cohesive Frictional*
BPM	Modelo de contato *Bonded-Particle Model*
PFC2D/3D	*Sofware* comercial de modelagem via DEM
GSA	Método de otimização global *Generalized Simulated Annealing*

Microparâmetros

E_m	Módulo de Young ou módulo elástico microscópico
k_n	Rigidez normal microscópica
t	Resistência à tração microscópica
k_s	Rigidez cisalhante microscópica
c	Coesão microscópica
ϕ_b	Ângulo de atrito local microscópico
ϕ_c	Ângulo de atrito residual microscópico

Macroparâmetros

C_0	Resistência à compressão simples (UCS)
E	Módulo de Young
ν	Razão de Poisson
ϕ	Ângulo de atrito

GSA

$g_{q_V}(\delta\vec{r})$	Distribuição de Tsallis
$P(\vec{r}_{t+1} \rightarrow \vec{r}_t)$	Probabilidade de aceitação
$T^V_{q_V}(t)$	Função de redução da temperatura
q_V	Parâmetro de controle da distribuição de Tsallis
q_A	Parâmetro de controle da probabilidade de aceitação

Variáveis

L	Largura da amostra
H	Altura da amostra
r_i ou d_i	Raio/diâmetro dos grãos
$r_{\max}/r_{\min}$	Razão entre o raio máximo e mínimo dos grãos
ρ	Densidade do material
v_l	Velocidade de carregamento
λ_d	Fator de amortecimento numérico
γ_{int}	*Interaction range*

Curvas

σ_1, σ_2 e σ_3	Tensões principais aplicadas ao material
σ_d	Tensão desviadora
ϵ_a	Deformação axial
ϵ_V	Deformação volumétrica

Função Objetivo

R_2	Fator de confiabilidade ou *reliability factor*

PUC-Rio - Certificação Digital Nº 1712747/CA

PUC-Rio - Certificação Digital Nº 1712747/CA

Sol, meu querido Sol, ilumina minha praia
Brilha, brilha o dia inteiro que eu não canso de brincar
Quando for de tardezinha que o senhor for indo embora
Me prometa que amanhã bem cedo você vai voltar

Mundo Bita, *Chuá Tchibum.*

1
Introdução

A simulação está presente em praticamente todos os campos do desenvolvimento humano e pode ser de elaborada de diversas maneiras. Atualmente está arraigado no inconsciente popular a sua relação com computação e avaliações numéricas. Por conta da sua alta reprodutibilidade, esta é, muitas vezes, a maneira mais barata e controlável de se tentar reproduzir um fenômeno no qual se deseja estudar.

Para a simulação mecânica de rochas ou materiais sólidos existem algumas técnicas que são bastante empregadas para a avaliação de relações de tensão-deformação. Entre elas estão o Método de Elementos Finitos (*Finite Element Method* - FEM) [1], além de diversas técnicas sem uso de malhas (*meshless* ou *mesh free*) [2, 3, 4] que tratam o material como um corpo contínuo e apresentam leis de tensão-deformação consistentes. Contudo estas aproximações contínuas não conseguem reproduzir, de maneira simples, a ocorrência de fraturas que envolvem altas quantidades de descontinuidades. O Método de Elementos Discretos (*Discrete Element Method* - DEM) [5] pode ser um alternativa para estudos deste tipo de fenômeno, primariamente por levar naturalmente em conta as descontinuidades.

Por conta do aumento do poder computacional, barateamento das suas peças e melhorias nas linguagens de programação ocorridos ao longo dos últimos anos, o DEM tem recebido uma maior atenção por apresentar uma grande aplicabilidade em inúmeras áreas da Engenharia.

O DEM consiste em solucionar as equações do movimento para cada consistituinte, avaliando o comportamento de todo o material. As forças entre os elementos são dadas por um modelo de contato e são estes modelos que descrevem a coesão do material. Diferentemente dos métodos contínuos, cujos parâmetros de entrada que representam o material são aqueles que descrevem o *bulk*, no DEM é necesário incluir os parâmetros que descrevem a interação em microescala, os chamados microparâmetros, entre os seus elementos constituintes. Cada um deles manifesta uma macroresposta distinta e, de maneira geral, não é possível se expressar, por uma única equação, a relação direta entre os microparâmetros à sua resposta macroscópica que se aplique a todos os casos, geometrias e modelos de contato.

PUC-Rio - Certificação Digital Nº 1712747/CA

Os diferentes modelos de contato podem incluir diferentes microparâmetros. Cada um apresenta aplicabilidade mais adequada a certos problemas como, por exemplo, mecânica de rochas ou solos, movimentação de blocos, grãos, etc. Por conta desta variedade de aplicações e da impossibilidade de introduzir os parâmetros macroscópicos desejados, um procedimento inicial inconveniente, porém de grande importância, é o chamado de calibração.

A calibração é um passo no qual se busca obter o conjunto de microparâmetros que resulte na resposta macroscópica desejada. Este é um processo lento e computacionalmente dispendioso em que normalmente é utilizada uma técnica ineficiente conhecida como tentativa e erro [6, 7, 8, 9, 10, 11]. Desta maneira, se faz necessário algum tipo de automatização. Uma forma de se realizar esta automatização é usando algum método de otimização, ou seja, de minimização de uma função objetivo. Sendo assim, o *Generalized Simulated Annealing* (GSA) [12] foi escolhido por apresentar algumas características interessantes que serão discutidas posteriormente.

Esta dissertação está dividida em seis capítulos. O capítulo 1 apresenta esta introdução; o capítulo 2 é uma revisão dos conceitos do DEM, do modelo de contato utilizado e da função objetivo que se deseja minimizar. Toda a discussão sobre otimização via GSA foi uma atualização do material escrito na minha dissertação de mestrado em Física [13]; o capítulo 3 descreve como foi realizado o modelo, quais parâmetros utilizados, as suas justificativas e quais tipos de ensaios executados; no capítulo 4 são discutidos os resultados; no capítulo 5 constam as conclusões do trabalho, enquanto o capítulo 6 são apresentadas as sugestões para trabalhos futuros.

PUC-Rio - Certificação Digital Nº 1712747/CA

2
Revisão Bibliográfica

O Método de Elementos Discretos foi proposto originalmente por Cundall e Strack[5] e consiste em representar o material como sendo composto de pequenas unidades interagentes que são chamadas de partículas ou elementos discretos. Neste contexto, as partículas têm uma conotação distinta da comumente utilizada na Física, cujas dimensões são desprezíveis quando comparadas com o todo.

Trata-se de um método bastante versátil, podendo ser utilizado nos mais variados campos da Ciência e Engenharia como, por exemplo, em modelagens de ancoragem em solos [7, 14], ensaios mecânicos e comportamento de rocha dura [15]. Na indústria do petróleo é usado na modelagem de *breakouts* em poços [6] ou fluxo de fluidos em meios porosos [16]. É também extensamente utilizado para modelar danos e comportamentos não lineares em materiais granulares [17].

Os seus constituintes são rígidos e mais comumente são utilizadas esferas, porém podem-se encontrar outras geometrias mais angulosas, tais como longarinas (Figura 2.1(a)) [14], *clumps* (Figura 2.1(b)) [15], elipsóides, cilindros e cubos (Figura 2.1(c)) [18] ou geometrias superquadráticas (Figura 2.1(d)) [19].

Diferentemente do Método de Elementos Finitos, cujos parâmetros de entrada são as respostas macroscópicas do material simulado, no caso do DEM necessita-se da inclusão de parâmetros que dizem respeito às interações entre os elementos discretos. Estes microparâmetros são ajustáveis ou, como mais comumente é conhecido na literatura, calibráveis. Como não existe uma equação geral que relacione diretamente estes microparâmetros e sua resposta macroscópica, necessita-se buscar valores destes microparâmetros cujas respostas sejam de interesse do modelador. Isto torna a calibração claramente um problema de otimização.

2.1
DEM

De uma maneira geral o DEM envolve cinco etapas:

1. Inicia-se um ciclo com a determinação do passo de tempo, que pode ser dinâmico, via cálculo de propagação de onda compressional, ou estático,

PUC-Rio - Certificação Digital Nº 1712747/CA

um valor fixo em toda a simulação;

2. Aplicação das leis do movimento ou, no caso, a segunda lei de Newton,

$$\sum \vec{F} = m\frac{\mathrm{d}^2\vec{r}}{\mathrm{d}t^2}, \tag{2-1}$$

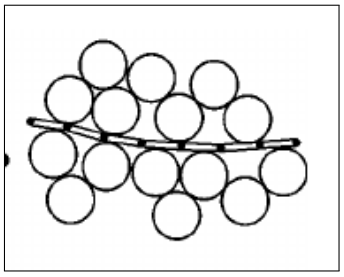
para cada um dos elementos do modelo. Uma vez que se tenham as forças, calcula-se as acelerações, velocidades e, por fim, as novas posições realizando integrações sucessivas. Este seqüência é feita para translação assim como para a rotação;

3. Acréscimo do passo de tempo;

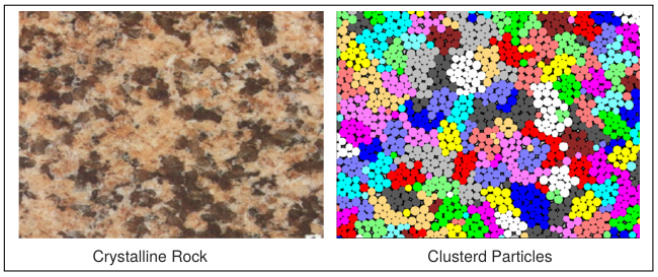
4. Detecção dos contatos, sendo, normalmente, um dos passos mais custosos computacionalmente;

5. A partir das novas posições, calculam-se as novas forças em cada elemento.

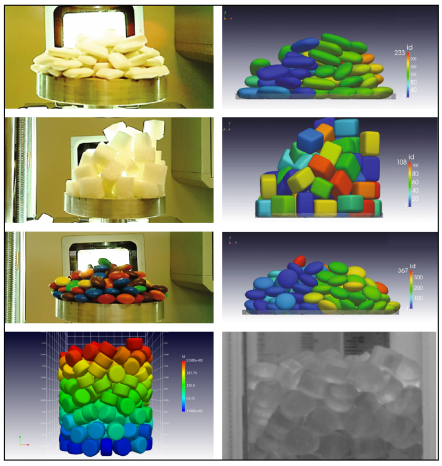
PUC-Rio - Certificação Digital Nº 1712747/CA



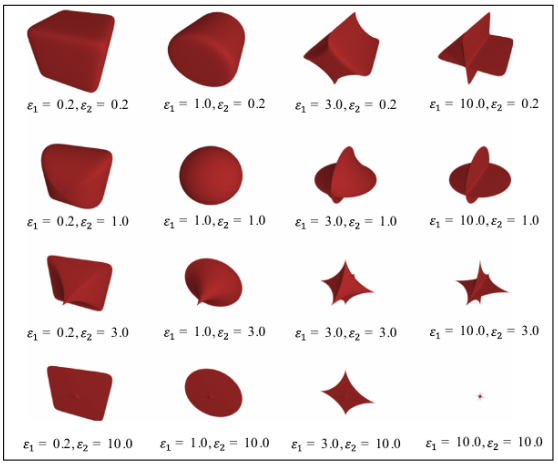
2.1(a): Geometria alongada. Conforme referência [14].



2.1(b): Aglomerado de esferas ou *clumps*. Conforme referência [15].



2.1(c): Geometrias elipsoidal, cúbica e cilíndrica. Conforme referência [18].



2.1(d): Geometria tridimensional superquadrática. Os parâmetros ϵ_1 e ϵ_2 são os definidores de forma dos elementos. Conforme referência [19].

Figura 2.1: As diversas geometrias utilizadas em simulações via DEM.

Este ciclo se repete até que um critério de parada seja atingido. A Figura 2.2 representa um ciclo de cálculos básico em DEM.

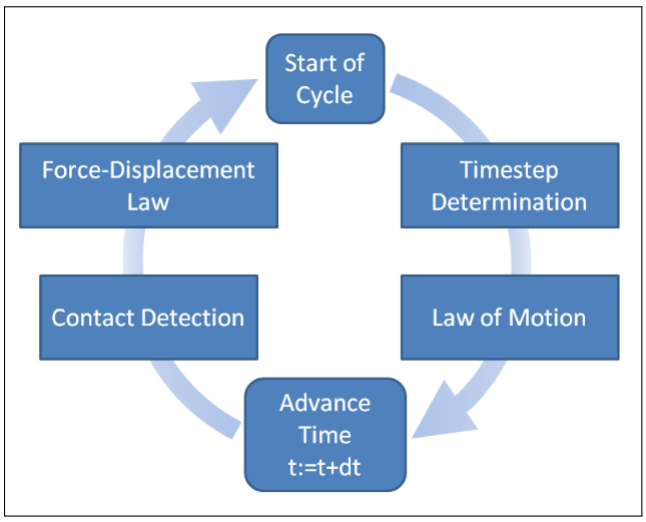


Figura 2.2: O esquema geral de cálculo em DEM, definido na referência [20].

Em rochas reais uma parte da energia é dissipada por fricção interna dos grãos. Então, na simulação de fenômenos quase estáticos, deseja-se que esta energia cinética dos elementos seja dissipada de maneira similar. Uma vez que a maioria das leis constitutivas, ou modelos de contato, não incluem o amortecimento baseado na velocidade, é possível incluir um amortecimento numérico artificial. A idéia é reduzir as forças forçadamente à medida em que há aumento das velocidades dos elementos por um fator ΔF_d. Esta redução é realizada por componente independentemente, o que torna este amortecimento um ente sem sentido físico. Não é invariante com respeito à rotação do sistema de coordenadas e é bastante simples de se calcular, necessitando somente como dado de entrada o valor do parâmetro de amortecimento λ_d. No Yade, o *software* utilizado neste trabalho, implementou-se uma versão que pode ser escrita da forma [21]

$$\Delta\vec{F}_d = -\lambda_d \vec{F}(t)\ \mathrm{sgn}\left[\vec{F}(t)\cdot\left(\frac{\mathrm{d}\vec{r}}{\mathrm{d}t}+\frac{\mathrm{d}^2\vec{r}}{\mathrm{d}t^2}\frac{\Delta t}{2}\right)\right], \tag{2-2}$$

onde $\vec{F}(t)$ é a força resultante, sgn é o sinal da operação entre as colchetes e $\vec{u}$ é a posição em cada elemento. O Δt é o passo de tempo e tem um valor máximo, chamado tempo crítico, para que o esquema das diferenças finitas produza uma solução estável.

PUC-Rio - Certificação Digital Nº 1712747/CA

2.1.1 Modelo de Contato

Os modelos de contato são responsáveis pela coesão observada nas rochas e cada modelo apresenta um conjunto de microparâmetros distintos que devem ser cuidadosamente escolhidos para simular melhor o comportamento de um material. O processo da seleção, teste e avaliação do conjunto de microparâmetros com o objetivo de se atingir determinada macro-resposta chama-se calibração.

Existem inúmeros modelos de contato com diferentes objetivos que descrevem os materiais mais variados, desde areia, sementes, até materiais coesivos como solos e rochas. Cada um apresenta um conjunto de microparâmetros distintos. Como exemplo, o trabalho de Peng[22] apresenta uma lista de modelos de contato utilizados para simular pavimentação.

Para simulação de rochas os modelos mais comuns são o *Bonded-Particle Model* (BPM) [10], disponível no *software* PFC3D [20], e o *Jointed Cohesive Frictional* (JCF) [9] codificado no *software* Yade [21] e utilizado neste trabalho. Este último modelo consegue representar uma rocha intacta ou fraturada por meio de indicação de diferentes microparâmetros.

Uma vez que a amostra do material a ser simulado é gerada, os pares dos elementos discretos são identificados por uma faixa de interação (*interaction range* - γ_{int}) dada por

$$D_e = \gamma_{\text{int}} \left(r_1 + r_2 \right), \tag{2-3}$$

onde D_e é a distância de equilíbrio, r_1 e r_2 são os raios dos elementos 1 e 2, respectivamente, e $\gamma_{\text{int}} \geq 1$.

Diferentemente do DEM clássico, no qual os contatos considerados são somente aqueles elementos interagentes, o número médio de ligações, também chamado de número de coordenação, é controlado pelo parâmetro γ_{int} antes do primeiro passo do ciclo computacional. Isto permite um aumento do grau de interligação sem as dificuldades computacionais relacionadas à lógica de *clumps*. Uma ilustração de como funciona o parâmetro γ_{int} pode ser vista na Figura 2.3. Uma vantagem desta abordagem é a possibilidade de se retornar à descrição do DEM clássico somente atribuíndo o valor $\gamma_{\text{int}} = 1$.

A força entre os elementos é decomposta em duas direções: normal e cisalhante. Na direção normal a força é dada pela equação

$$F_n = k_n u_n \tag{2-4}$$

onde k_n é a rigidez normal e u_n é o deslocamento normal entre dois elementos interagentes, definido por $u_n = D - D_e$, com D sendo a distância entre os centros das esferas. O k_n ainda pode ser escrito como

PUC-Rio - Certificação Digital Nº 1712747/CA

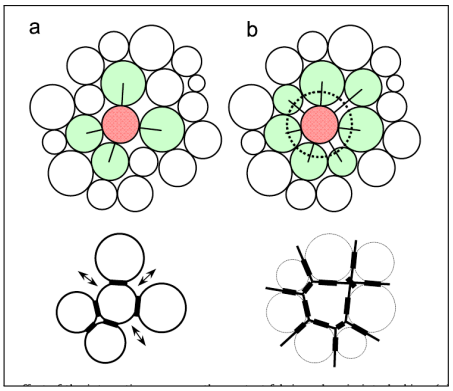


Figura 2.3: Representação geométrica do parâmetro γ_{int}. Em (a) representa-se $\gamma_{\text{int}} = 1$ no qual somente os elementos que se tocam estão ligados, enquanto em (b) representa-se $\gamma_{\text{int}} > 1$, onde uma faixa de interação é levada em consideração, aumentando-se assim o número de coordenação. Conforme referência [8].

$$k_n = E_m \frac{r_1 r_2}{r_1 + r_2} \tag{2-5}$$

onde E_m é chamado de módulo de Young ou módulo elástico. Sob tração a ruptura acontece quando a força atinge o valor máximo dado por

$$F_{n,\max} = -tA_{\text{contato}}, \tag{2-6}$$

onde t é chamado de resistência à tração e a área de contato $A_{\text{contato}} = \pi \min(r_1, r_2)^2$. A partir do momento que há a ruptura, a força normal é zerada. A Figura 2.4 ilustra a força normal entre dois elementos.

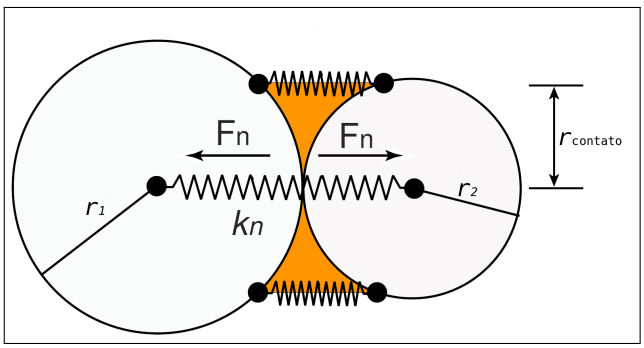


Figura 2.4: Representação da força normal em dois elementos discretos. Adaptado da referência [15].

A força cisalhante total, que pode ser vista na Figura 2.5, é calculada de maneira incremental, atualizando a sua orientação e intensidade por um incremento da força cisalhante, ou seja,

$$F_s = \{F_s\}_{\text{anterior}} + k_s \Delta u_s \tag{2-7}$$

PUC-Rio - Certificação Digital Nº 1712747/CA

com k_s sendo a rigidez cisalhante e Δu_s é deslocamento tangencial incremental relativo.

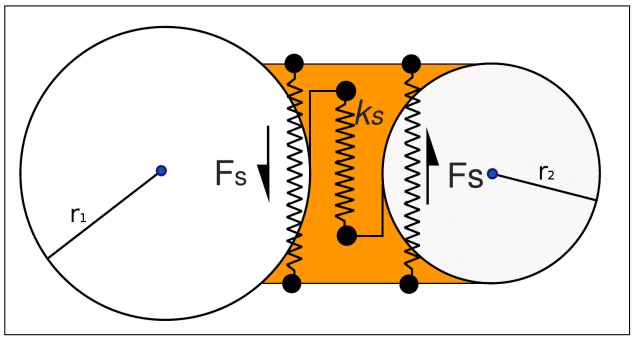


Figura 2.5: Representação da força cisalhante entre dois elementos discretos. Adaptado da referência [15].

Para modelar o comportamento não linear da relação entre tensão e deformação entre os elementos, utilizou-se um modelo de Mohr-Coulomb modificado. A força de cisalhamento máxima é dependente da força normal (F_n), coesão (c), ângulo de atrito local (ϕ_b) e ângulo de atrito residual (ϕ_c). A força de cisalhamento máxima é calculada de acordo com

$$F_{s,\max} = F_n \tan \phi_b + cA_{\text{contato}}. \tag{2-8}$$

Após a ruptura, os contatos são considerados puramente friccionais e são definidos por

$$F_s = F_n \tan \phi_c \tag{2-9}$$

Uma representação das relações entre F_n e F_s pode ser vista na Figura 2.6.

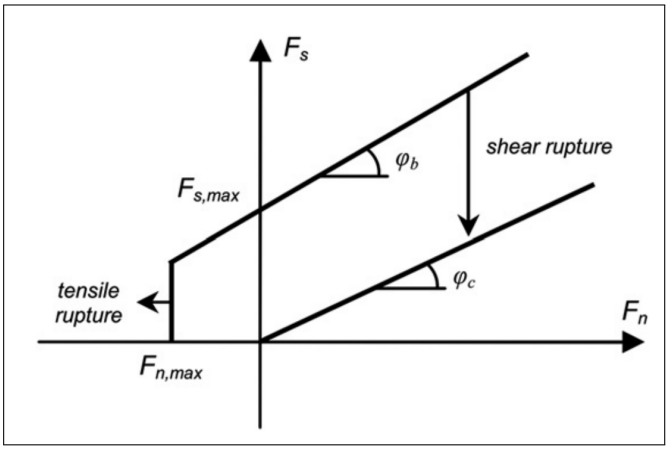


Figura 2.6: Modelo de Mohr-Coulomb modificado conforme modelo de contato JCF. Após a ruptura, não há mais coesão e há a mudança do ângulo de atrito de ϕ_b para ϕ_c. Conforme referência [9].

PUC-Rio - Certificação Digital Nº 1712747/CA

2.2 Uso de Otimização em DEM

Ao se trabalhar com simulação em DEM, um dos passos mais importantes é a calibração dos microparâmetros de maneira que as respostas macroscópicas sejam os valores desejados de materiais reais que se deseje simular. O processo de calibração foi dividido por Coetzee[23] em duas vertentes: a primeira usa como dados de entrada as medidas físicas diretas das propriedades microscópicas. Esta abordagem, chamada de *direct measurement*, tem a vantagem de ser independente do modelo de contato a ser utilizado, porém apresenta dificuldades na obtenção dos valores de certas propriedades, principalmente à medida que os grãos tendem a ficar menores e mais angulosos. A outra abordagem, conhecida como *bulk calibration*, trata-se da modificação dos valores dos microparâmetros de maneira a reproduzir numericamente o comportamento geral da rocha em experimentos de laboratório. Para isto é necessária a simulação de vários experimentos.

Dentro deste segundo grupo, o método mais usado, incrivelmente, é o mais inecifiente: a tentativa e erro [6, 7, 8, 9, 10, 11]. Para um modelador experiente, torna-se menos complicado determinar o conjunto de microparametros que mais se aproxima dos resultados macroscópicos desejados. Porém esta é uma técnica que escala exponencialmente com o aumento da quantidade de variáveis usadas. Isto dificulta muito o processo como um todo por ser intensamente tedioso e computacionalmente desgastante. Além disso, o processo de calibração é fortemente influenciado pelo modelo de contato utilizado.

Alguns autores propuseram formas de automatização deste processo. Sem nenhuma ordem específica pode-se destacar o trabalho de Nguyen, André e Huger[24] que propõe uma calibração usando funções que relacionam os microparâmetros com os macroparâmetros. Estas funções são obtidas por mínimos quadrados não linear e os dados numéricos são gerados a partir da simulação de 1600 a 6400 ensaios de tração com três variáveis.

O trabalho de Yoon[25] utilizou uma nova maneira de calibrar os microparâmetros de modelos *contact bond* no PFC2D. Porém concluiu-se que o método pôde somente determinar os microparâmetros de rochas com suas propriedades mecânicas dentro das faixas de UCS (C_0) entre 40 MPa e 170 MPa, módulo de Young (E) entre 20 GPa e 50 GPa e razão de Poisson (ν) entre 0,19 e 0,25, ou seja, esta proposta tem uma aplicação restrita. Apesar desta limitação, o método é bastante interessante pois envolve uma análise de sensibilidade para determinar quais são os microparâmetros que têm mais influência na resposta macroscópica, determinando-se as relações não lineares via *Central Composite Design* e por fim realizando-se a otimização das fun-

PUC-Rio - Certificação Digital Nº 1712747/CA

ções com restrições. Calibraram-se sete microparâmetros somente com ensaio uniaxial objetivando-se os valores das propriedades mecânicas citadas.

Deng et al.[26] aplicaram uma metodologia semelhante para estudar a transferência de calor via DEM. Utilizaram-se três microparâmetros sob três forças externas em 15 simulações. Já Chehreghani et al.[27] usaram uma variação destes trabalhos para calibrar cinco microparametros do modelo de contato BPM, atingindo-se os valores determinados de C_0 e E em 32 simulações de ensaio uniaxial.

No trabalho de Rackl e Hanley[28] usou-se simultaneamente duas técnicas. A primeira, com o divertido nome de *latin hypercube*, gera pontos aleatórios num espaço N-dimensional seguido da krigagem, que é uma técnica estatística para estimar valores da função de uma variável aleatória na região entre pontos fixados. Calibraram-se seis variáveis com uma média de 51 simulações.

Os pesquisadores Do, Aragón e Schott[29] propuseram o uso de duas técnicas de otimização diferentes: o algoritmo genético, que é um método de otimização inspirado na teoria da evolução das espécies e trata as variáveis como cromossomos que, por meio de operadores de recombinação, elitismo e mutação, selecionam-se as melhores soluções conforme um critério de convergência; e uma variação do método de Lipchizian chamado DIRECT (*DIviding RECTangles*) que realiza otimizações simultâneas nos níveis local e global por um processo iterativo que divide o espaço de busca em hiper-retângulos com uma única amostra no seu centro. Realizou-se a otimização em duas variáveis objetivando-se três macroparâmetros. Via algoritmo genético conseguiu-se a convergência com aproximadamente 30 simulações e o DIRECT com 275 simulações. No trabalho seguinte [30] propuseram-se o uso de uma variação do algoritmo genético chamado *Non-dominated Sorting Genetic Algorithm II*, que visa trabalhar melhor a otimização de funções multidimensionais com multiobjetivos. Neste trabalho otimizou-se cinco variáveis objetivando-se três macroparâmetros em 30 gerações.

Mais recentemente Simone, Souza e Roehl[31] propuseram o uso do algoritmo genético para o processo de calibração, usando o software Yade com o modelo de contato tipo JCF, simulando o ensaio uniaxial e com função objetivo os valores macroscópicos E e C_0. As variáveis utilizadas são todos os microparâmetros deste modelo de contato (E_m, k_s/k_n, ϕ_b, c e t). Neste trabalho avaliou-se a influência da população inicial, que possibilitou escrever a equação que relaciona os microparâmetros com cada um dos macroparâmetros avaliados. Utilizando-se somente o ensaio uniaxial percebeu-se que os valores de E e C_0 sofrem as maiores influências das variáveis E_m, c e t, e pouca influência de ϕ_b e k_s/k_n.

PUC-Rio - Certificação Digital Nº 1712747/CA

Já Tawadrous et al.[32] conduziram calibrações utilizando-se redes neurais, que são um conjunto de técnicas matemáticas que usam como analogia a interação biológica entre neurônios. Calibraram-se cinco variáveis e três macroparâmetros (E, ν, C_0) via ensaio uniaxial. Contudo, ao aumentar o número de variáveis para seis, não foi possível calibrar os macroparâmetros adequadamente. A principal chave deste método é um número suficientemente grande de dados de treinamento, que implica em um grande número de simulações. Para este caso, utilizou-se 3025 para treinar a rede e mais 100 para calibrar. De maneira semelhante Benvenuti, Kloss e Pirker[33] apresentaram um estudo com cinco variáveis visando-se quatro macroparâmetros.

Fakhimi e Villegas[34] utilizaram-se da análise dimensional para otimizar sete variáveis via ensaios uniaxial e brasileiro, com E e ν como objetivos.

O único trabalho que se utilizou de alguma variação do *Simulated Annealing* para calibração dos microparâmetros em DEM, foi o artigo dos pesquisadores Wang e Cao[35]. Utilizam-se o *Fast Simulated Annealing* (que os autores nomearam por algum motivo de *Improved Simulated Annealing*) para otimizar nove variáveis, com os macroparâmetros E, ν e C_0 como objetivo, via 2739 simulações de ensaio uniaxial. Apesar deste trabalho apresentar alguns pontos que podem ser questionados, como por exemplo, faixa de variação dos microparâmetros, tamanho da cadeia de Markov e temperatura inicial exageradamente altas, a ausência ou omissão do número de otimizações realizadas até atingir a convergência, cálculo do desvio padrão de cada variável e cada macroparâmetro e a não apresentação das curvas tensão-deformação e deformações volumétrica-axial finais, foi este artigo que motivou o presente trabalho por apresentar uma primeira aplicação de uma versão do *Simulated Annealing* na a calibração em DEM.

PUC-Rio - Certificação Digital Nº 1712747/CA

2.3 Generalized Simulated Annealing

Muitos problemas da Ciência estão relacionados com a minimização ou maximização de uma determinada função objetivo em um espaço multidimensional. Para resolver este tipo de problema muitas vezes se faz necessária a introdução de métodos numéricos que, por características do próprio hiperespaço, buscam a convergência para a solução desejada.

Desde o início da pesquisa na área, várias alternativas foram propostas para a minimização de uma função objetivo. Os vários métodos podem ser classificados em dois tipos:

1. Métodos baseados em uma busca do tipo tentativa e erro, também conhecido como busca exaustiva:

São métodos que demandam muito tempo e esforço por não conterem uma metodologia de exploração do hiperespaço. Idealmente só seriam utilizados nos casos em que se faz necessária uma investigação muito superficial, utilizando-se poucas variáveis e na região próxima ao ponto de origem.

2. Métodos baseados em uma rotina sistemática para minimização da função objetivo:

   Dividem-se em basicamente duas classes que compõem os **métodos de busca local**, que utilizam ferramentas essencialmente numéricas para localizar um mínimo nas proximidades do ponto de origem, sem se preocupar com a "profundidade" deste mínimo; e os **métodos de busca global**, que visam explorar, de forma estatística, todo o hiperespaço de parâmetros, ou a parte dele que faz sentido físico, sendo assim em princípio capazes de distinguir entre os vários mínimos locais e o mínimo global.

PUC-Rio - Certificação Digital Nº 1712747/CA

Dos inúmeros métodos que realizam a busca global podem ser destacados os algoritmos genéticos [36], DIRECT [37, 38], *Controlled Random Search* [39], *Multi-Level Single-Linkage* (MLSL) [40, 41], *Particle Swarm Optimization* (PSO) [42] e o *Simulated Annealing* [43, 44, 45, 12, 46]. O código fonte de alguns destes algoritmos pode ser encontrado no GitHub da Nonlinear Optimization (NLOPT) [47].

O *Simulated Annealing* tem atraído uma relativa atenção por ser adequado em tratar problemas de minimização quando estão envolvidas muitas variáveis e o mínimo global está compreendido em um conjunto com vários outros mínimos de menor profundidade. As principais características que contribuem para este fato são: ser *derivative free*, ou seja, não se necessita o cálculo das derivadas da função a ser otimizada; a sua relativa facilidade de incorporação em códigos que necessitem de uma otimização; e pela capacidade de ser empregado com basicamente nenhuma informação sobre a natureza específica do problema.

O método *Simulated Annealing* original foi proposto independentemente por Kirkpatrick, Gelatt e Vecchi[43] e Černý[44] no início da década de 1980 para a solução do problema do caixeiro viajante. Foi inspirado em processos metalúrgicos no quais os metais em estado líquido são gradualmente resfriados. Em altas temperaturas os átomos movem-se em completa desordem em todas as direções. Porém, à medida que a temperatura do sistema é lentamente reduzida, estes átomos tendem a perder a mobilidade, se agrupando em direções preferenciais, possibilitando-se assim a formação de uma rede cristalina. Este

cristal corresponde ao estado de energia mínima do sistema ou o mínimo global de energia termodinâmica. Por outro lado, caso a temperatura do sistema seja reduzida mais drasticamente e, dependendo das condições físicas prevalecentes durante o processo de resfriamento, os átomos não terão tempo para se organizar de modo que o sistema torna-se um sólido policristalino ou amorfo. A este estado termodinâmico chama-se de mínimo local.

Um outro exemplo deste fenômeno pode ser encontrado na Geologia, na qual as rochas plutônicas, ou ígneas intrusivas, são formadas dentro da crosta terrestre por um lento resfriamento do magma, dando origem a rochas com cristais bem definidos como no granito ou gabro; e também as rochas vulcânicas, ou ígneas extrusivas, que são geradas por um resfriamento bastante rápido na superfície ou na água, não dando tempo para a cristalização, como no caso do basalto ou da pedra-pomes.

De uma maneira abrangente, o *Simulated Annealing* e suas atualizações funcionam da seguinte maneira: partindo-se de um ponto $\vec{r}_t$ no hiperespaço de variáveis, um novo ponto será dado a partir da adição de um incremento aleatório, $\delta\vec{r}$, calculado mediante uma determinada distribuição de probabilidade. A partir deste novo ponto, $\vec{r}_{t+1} = \vec{r}_t + \delta\vec{r}$, calcula-se a função objetivo e se efetua a comparação com o valor no ponto anterior. Uma vez que esta diferença seja negativa, o que indica um movimento de descida, o movimento é automaticamente aceito, substituindo-se o ponto $\vec{r}_t$ pelo ponto $\vec{r}_{t+1}$, retornando-se ao processo. Se a diferença for positiva, o que significa um movimento de subida no valor da função objetivo, o movimento pode ou não ser aceito, conforme a probabilidade dada por um certo critério de aceitação, ou seja, $P(\vec{r}_{t+1} \rightarrow \vec{r}_t)$.

Por conta desta probabilidade de um movimento de subida ser aceito, o algoritmo consegue sair de um mínimo local. Caso este movimento não seja aceito, o novo ponto é rejeitado e calcula-se novamente o valor de $\delta\vec{r}$ repetindo-se o processo. Após uma quantidade determinada de vezes que este processo é repetido, reduz-se a temperatura conforme uma função $T^V_{q_V}(t)$. Neste contexto, a temperatura é uma medida da intensidade da aleatoriedade do sistema, ou seja, à medida que o método evolui, a temperatura reduz, diminuindo o seu nível de aleatoriedade. Um fluxograma de todo este processo pode ser visto na Figura 2.7.

Com o passar dos anos algumas variações do *Simulated Annealing* foram propostas visando melhorar o seu desempenho. Em 1987, os pesquisadores Szu e Hartley[45] propuseram a modificação da função distribuição da gaussiana para a de Cauchy-Lorentz e, como conseqüência, a redução da temperatura poderia ser mais rápida. Esta versão ficou conhecida como *Fast Simulated Annealing*.

PUC-Rio - Certificação Digital Nº 1712747/CA

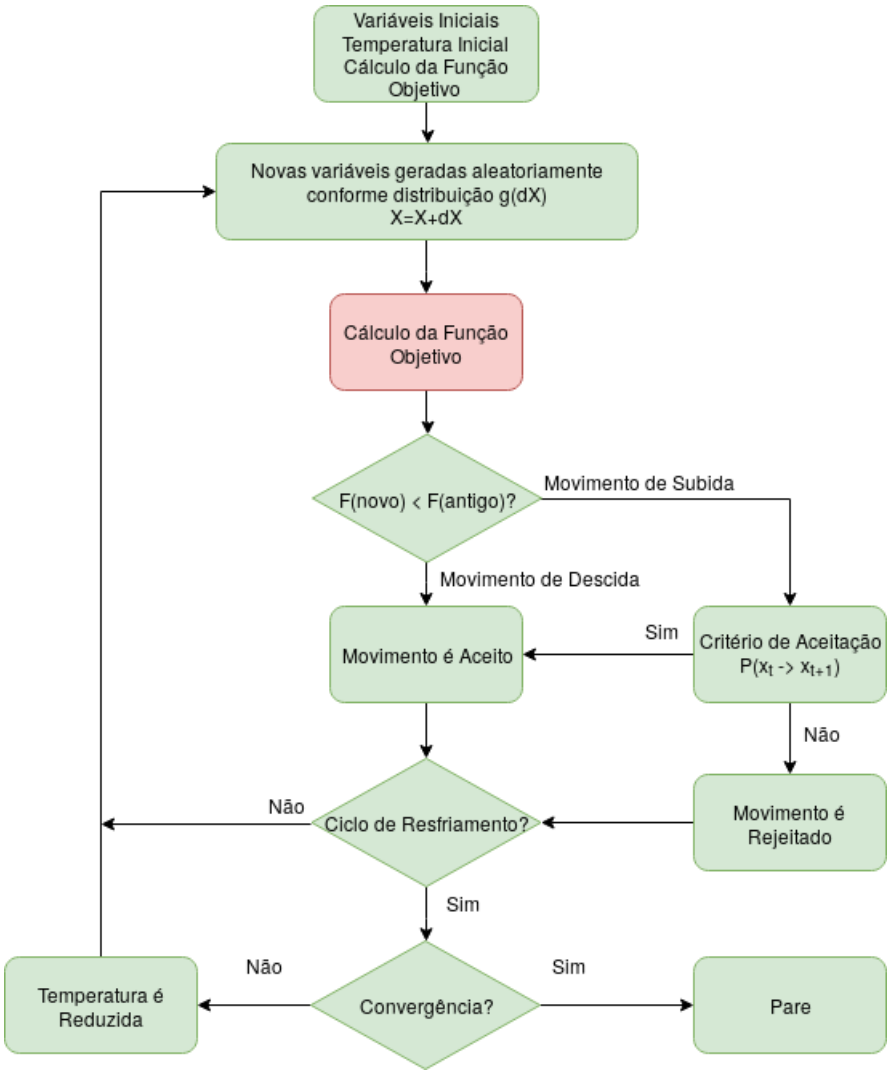


Figura 2.7: Fluxograma do *Simulated Annealing*. O passo em destaque é onde se insere a função objetivo, conforme o problema proposto. A série de cálculos efetuados com uma mesma temperatura é chamada de cadeia de Markov.

Já em 1996, Tsallis e Stariolo[12] publicaram a versão generalizada com dois parâmetros de controle, o *Generalized Simulated Annealing* (GSA). O primeiro, q_V, define a forma da função distribuição e a redução da temperatura, enquanto o q_A relaciona-se com a probabilidade de aceitação.

Mais recentemente Andrade, Mundim e Malbouisson[46] publicaram a versão mais generalizada usando três parâmetros de controle, q_V, q_A e q_T, que defininem a função distribuição, a probabilidade de aceitação e a redução da temperatura, respectivamente. Nestes dois últimos casos, é possível recuperar as versões anteriores com determinados valores de q_V, q_A e q_T.

Como dito anteriormente, estas quatro versões têm seu funcionamento da mesma forma. O que as diferenciam são, basicamente, as equações que regem a função distribuição, probabilidade de aceitação e redução da temperatura.

Um dos pontos mais importantes do método é como os números aleatórios

PUC-Rio - Certificação Digital Nº 1712747/CA

são gerados eficientemente de maneira que obedeçam à distribuição desejada. Por conta desta situação utilizou-se, neste trabalho, a versão do GSA com dois parâmetros [12] que apresenta o algoritmo do gerador de números aleatórios mais maduro e estável. Os aspectos teóricos do GSA serão discutidos a seguir.

2.3.1 Função Distribuição ($g_{q_V}(\delta\vec{r})$)

A equação que rege a distribuição de probabilidade de Tsallis para o sorteio das novas variáveis tem o aspecto de uma distribuição de Cauchy-Lorentz distorcida. É determinada principalmente pelo parâmetro q_V e sua forma analítica é dada por

$$g_{q_V}(\delta\vec{r}) = \left(\frac{q_V-1}{\pi}\right)^{\frac{D}{2}} \cdot \frac{\Gamma\left(\frac{1}{q_V-1}+\frac{D-1}{2}\right)}{\Gamma\left(\frac{1}{q_V-1}-\frac{1}{2}\right)} \cdot \frac{[T^V_{q_V}(t)]^{-\frac{D}{3-q_V}}}{\left\{1+(q_V-1)\frac{(\delta\vec{r})^2}{[T^V_{q_V}(t)]^{\frac{2}{3-q_V}}}\right\}^{\frac{1}{q_V-1}+\frac{D-1}{2}}} \tag{2-10}$$

onde Γ representa a função gama, D é a dimensão do problema ou o número de variáveis utilizados na busca, q_V é um parâmetro que controla a abertura da função e $T^V_{q_V}(t)$ é a temperatura. Um exemplo de como a distribuição da Equação 2-10 varia com q_V pode ser visto no gráfico da Figura 2.8.

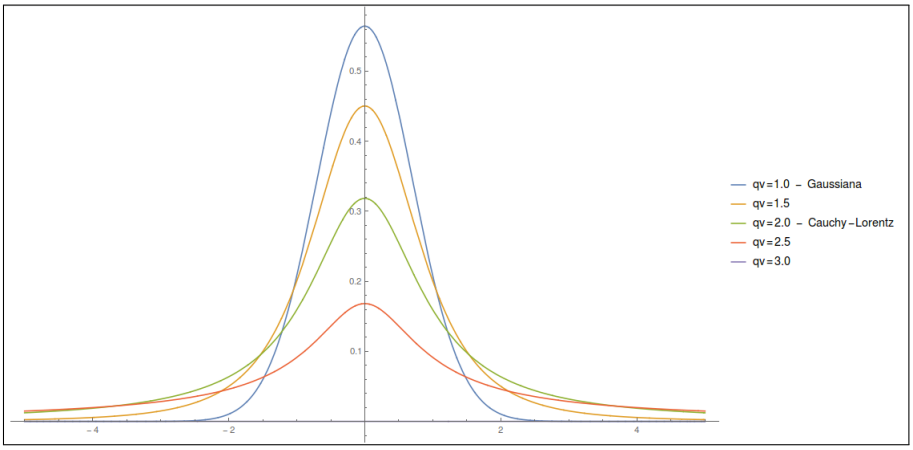


Figura 2.8: Gráfico da função distribuição de Tsallis para diversos valores de q_V normalizados com respeito à área sob as curvas. Este exemplo foi gerado com $D = 1$ e $T = 1$. Observa-se que, à medida em que os valores de q_V aumentam, a função torna-se cada vez mais uma distribuição de cauda longa, ou seja, uma maior probabilidade de haver sorteios mais distantes do centro da distribuição.

Um exemplo que ilustra a importância da abertura da distribuição na procura pelo mínimo global pode ser visto na Figura 2.9. Neste caso

PUC-Rio - Certificação Digital Nº 1712747/CA

pode-se verificar que as distribuições uniforme e gaussiana ($q_V = 1$) ficam concentradas em torno do mínimo menos profundo ou mínimo local (ponto A). Já a distribuição de Cauchy-Lorentz ($q_V = 2$), centrada no mesmo ponto A, consegue apresentar uma probabilidade não nula de se atingir o mínimo mais profundo ou o mínimo global (ponto B). Por conta desta probabilidade de se alcançar pontos mais afastadas, o algoritmo tem uma chance maior de escapar de mínimos locais, explorando uma região maior do hiperespaço.

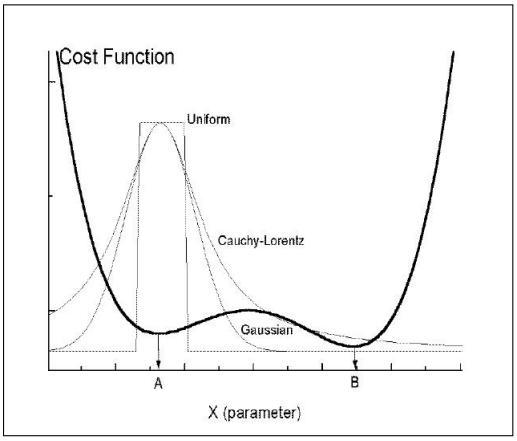


Figura 2.9: Representação unidimensional das distribuições uniforme, gaussiana ($q_V = 1$) e de Cauchy-Lorentz ($q_V = 2$) para uma mesma temperatura. Observa-se também que a função objetivo apresenta um mínimo local (ponto A) e um mínimo global (ponto B). Conforme referência [48].

PUC-Rio - Certificação Digital Nº 1712747/CA

2.3.2 Esquema de Resfriamento ($T^V_{q_V}(t)$)

À medida que o algoritmo evolui, reduz-se a temperatura, ou seja, a aleatoriedade, conforme uma determinada função de maneira a garantir a convergência. Esta função é definida por

$$T^V_{q_V}(t) = T_0 \frac{2^{q_V - 1} - 1}{(1+t)^{q_V - 1} - 1}, \tag{2-11}$$

onde T_0 é a temperatura inicial e q_V define o quão rápido será a redução da temperatura com o tempo[1]. A influência do parâmetro q_V na redução da temperatura com o tempo pode ser vista no gráfico da Figura 2.10. Percebe-se que para q_V mais altos há uma tendência da redução da temperatura ser mais severa, o que pode ocasionar dois resultados: a convergência ao mínimo global ser muito rápida ou o algoritmo ficar preso em um mínimo local.

[1]No contexto do GSA, o tempo representa a quantidade de cadeias de Markov.

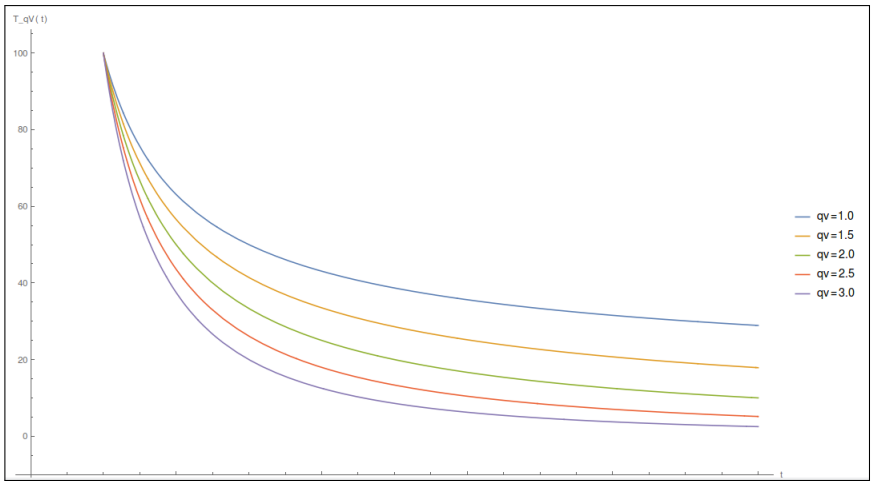


Figura 2.10: Gráficos do esquema de resfriamento, ou função temperatura generalizada, em função do tempo (ou número de cadeias de Markov), para vários valores de q_V. Este gráfico foi gerado com $T_0 = 100$. À medida em que se aumenta o valor do parâmetro q_V, o resfriamento tende a ser mais severo.

2.3.3 Probabilidade de Aceitação ($P_{q_A}(\vec{r}_t \to \vec{r}_{t+1})$)

Para haver a possibilidade de fuga de mínimos locais, deve-se ter passos em que a função objetivo aumente em relação ao valor anterior. Para isto é preciso definir a probabilidade de aceitação que é dada pela expressão

$$P_{q_A}(\vec{r}_t \to \vec{r}_{t+1}) = \begin{cases} 1 & , \text{ se } \Delta f < 0 \\ \dfrac{1}{\left[1+(q_A-1)\frac{\Delta f}{T^A_{q_A}}\right]^{1/(q_A-1)}} & , \text{ se } \Delta f \geq 0 \end{cases} , \tag{2-12}$$

onde q_A é um parâmetro que regerá o tipo de probabilidade de aceitação utilizada e $T^A_{q_A}$ é a temperatura. Não existe razão particular para que as funções $T^V_{q_V}(t)$ e $T^A_{q_A}(t)$ sejam iguais e para simplificar Tsallis e Stariolo[12] assumiram tal igualdade. Porém em trabalhos mais recentes [49] propôs-se a relação

$$T^A_{q_A}(t) = \frac{T^V_{q_V}(t)}{t} \tag{2-13}$$

que aumentou a eficiência do método. Utilizou-se este esquema neste trabalho.

A aceitação funciona da seguinte forma: sorteia-se um valor entre 0 e 1 com distribuição uniforme e compara-se com o valor calculado pela Equação 2-12. Caso o valor sorteado seja menor que a função $P_{q_A}(\vec{r}_t \to \vec{r}_{t+1})$, o passo de subida é aceito, caso contrário, descarta-se.

Um exemplo do comportamento da probabilidade de aceitação em relação à mudança do q_A pode ser vista nos gráficos da Figura 2.11. Uma vez fixo o q_A, para qualquer par de valores (aleatório entre 0 e 1 e Δf) abaixo da curva correspondente na Figura 2.11, o passo é aceito. Caso contrário, é rejeitado.

PUC-Rio - Certificação Digital Nº 1712747/CA

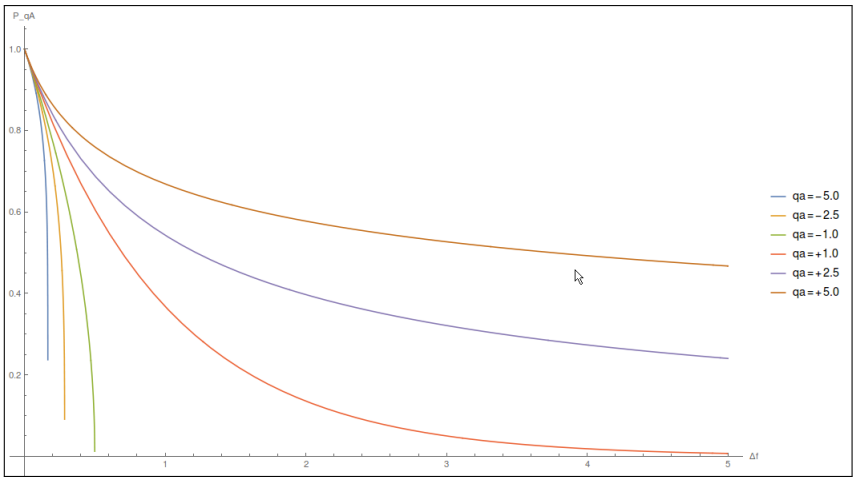


Figura 2.11: Gráfico da probabilidade de aceitação em função da diferença da função objetivo. Para valores de q_A menores há uma tendência a aceitar menos passos de subida, aumentando a eficiência.

Como comentado anteriormente, com a escolha correta dos pares (q_V, q_A) é possível recuperar as versões original ($q_V = 1$), que usa a distribuição gaussiana e a redução logarítmica da temperatura, e a versão FSA ($q_V = 2$), que usa a distribuição de Cauchy-Lorentz e a redução da temperatura inversamente proporcional ao tempo. No que diz respeito à probabilidade de aceitação, em ambos os casos, utiliza-se o critério de Boltzman, ou seja, $q_A = 1$.

2.4 Função Objetivo

Todos os trabalhos encontrados na literatura que envolvem calibração e DEM realizam-se várias simulações dos experimentos que se tentam reproduzir para encontrar os macroparâmetros desejados. Por exemplo, simulam-se ensaios de compressão e tração com objetivo de se obter os valores macroscópicos do módulo de Young, razão de Poisson, resistência à tração, etc. Contudo esta técnica tem alguns incovenientes, como exemplo, a necessidade de se otimizar uma função multi-objetivo e principalmente por não levar em consideração evolução temporal do ensaio. Pensando neste problema e inspirado nas questões de espalhamento dinâmico em difração de elétrons, utilizou-se como função objetivo o fator de confiabilidade (*reliability factor*) R_2 de LEED [50] definido por

$$R_2 = A_2 \int \left[I_e(x) - cI_c(x)\right]^2 \mathrm{d}x \tag{2-14}$$

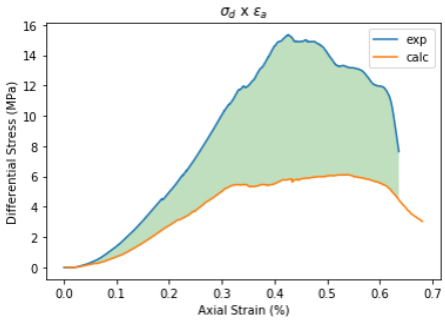
com A_2 sendo uma constante de normalização, de maneira que todas as curvas apresentem o mesmo peso no cômputo geral, sendo dada pela expressão

$$A_2 = \frac{1}{\int \left[I_e(x)\right]^2 \mathrm{d}x}. \tag{2-15}$$

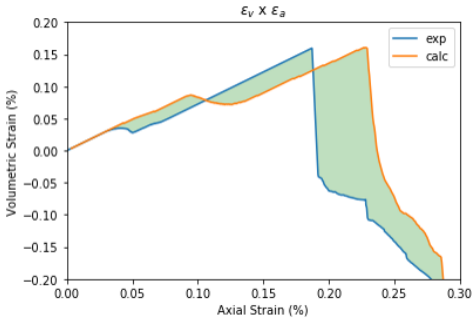
PUC-Rio - Certificação Digital Nº 1712747/CA

As funções $I_e(x)$ e $I_c(x)$ representam as intensidades das curvas experimentais e calculadas, respectivamente. A constante c é um fator de normalização da intensidade relativa. Como não é de interesse a intensidade relativa e sim a intensidade absoluta, esta constante foi feita igual a 1 em todas as simulações. Do ponto de vista matemático, a Equação 2-14 representa um funcional, pois as variáveis são as funções intensidade $I_e(x)$ e $I_c(x)$. Porém neste contexto os termos função objetivo, funcional, fator R_2 e fator de confiabilidade têm o mesmo significado.

Uma maneira simples de se visualizar o significado geométrico do funcional dado pela Equação 2-14 se dá por meio dos gráficos da Figura 2.12. O objetivo se torna minimizar a área quadrática normalizada entre as curvas experimentais e calculadas de $\sigma_d \times \epsilon_a$ e $\epsilon_V \times \epsilon_a$ em ensaios de compressão triaxial, de maneira que sejam as mais semelhantes entre si.



2.12(a): Gráficos de tensão *vs* deformação axial para dois ensaios uniaxial (experimental e calculado).



2.12(b): Gráficos da deformação volumétrica *vs* deformação axial para o mesmo ensaio.

Figura 2.12: O funcional R_2 representa a área quadrática normalizada entre as curvas experimental e calculada.

O funcional R_2 foi escolhido por ser amplamente utilizado em problemas de difração de elétrons e algumas aplicações de cristalografia de raios X, além de apresentar um mínimo evidente quando o integrando torna-se nulo. Uma demonstração do mínimo deste funcional pode ser visto no Anexo A.

PUC-Rio - Certificação Digital Nº 1712747/CA

3
Metodologia

O processo de calibração é um passo bastante importante em qualquer simulação via DEM. Para realizar a calibração é necessário modificar algumas variáveis e simular numericamente os experimentos com o objetivo de se obter sua resposta macroscópica. Neste capítulo descreve-se quais variáveis e/ou parâmetros foram utilizadas no DEM, no ensaio triaxial e no GSA, além dos motivos que levaram às suas escolhas.

3.1
Yade

O Yade [21] é um *software* de código aberto que realiza simulações de materiais através da técnica de DEM. Seu código inicialmente foi desenvolvido pelos pesquisadores da Universidade de Grenoble na França, mas atualmente conta com desenvolvedores espalhados pelo mundo inteiro. Seu núcleo foi desenvolvido em C++ com uma API em Python. Este detalhe deixa a curva de aprendizado bastante íngreme, porém torna o controle de toda a simulação muito mais simples e eficiente.

3.2
DEM

Por variáveis são chamados ou considerados todos aqueles parâmetros de entrada da simulação que têm ou podem ter influência nos valores dos macroparâmetros obtidos. Desta maneira, elas podem ser divididas em duas categorias: a primeira não diz respeito ao modelo de contato porém está relacionada com o DEM propriamente dito, sendo de extrema importância que sua escolha seja adequada. Pode-se destacar as variáveis:

- Dimensões da amostra (L e H);
- Raio dos grãos (r_i);
- Razão entre o raio máximo e mínimo dos grãos ($r_{\max}/r_{\min}$);
- Densidade (ρ);
- *Interaction range* (γ_{int});
- Velocidade de carregamento (v_l);

PUC-Rio - Certificação Digital Nº 1712747/CA

– Amortecimento numérico (λ_d).

Estas variáveis não foram utilizadas na otimização pois apresentam algumas dificuldades computacionais que serão discutidas a seguir. Desta maneira tomou-se valores constantes em todo o trabalho. Já o segundo grupo representa diretamente o modelo de contato utilizado, o JCF. No contexto deste trabalho, os microparâmetros são as variáveis que estão relacionadas diretamente com o modelo de contato e são:

– Coesão (c);
– Ângulo de atrito (ϕ_b);
– Ângulo de atrito residual (ϕ_c);
– Razão de Poisson (k_s/k_n);
– Resitência à tração (t);
– Módulo de Young (E_m).

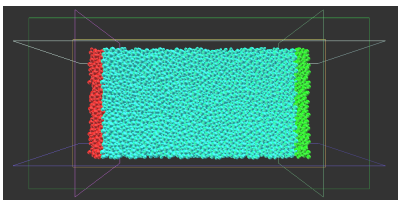
PUC-Rio - Certificação Digital Nº 1712747/CA

3.2.1 Dimensões das Amostras

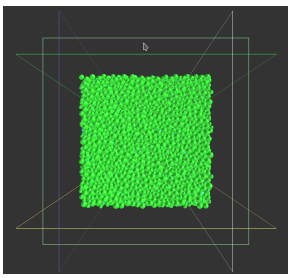
Para a construção da amostra foram seguidas as sugestões da *International Society of Rock Mechanics* [51] e da *American Society for Testing and Materials* [52], conforme visto na Tabela 3.1. A amostra tem a forma de um paralelepípedo reto com base quadrada de lado 44,3 mm e altura 88,6 mm, ou seja, $H = 2L$. As esferas têm raios com valores aletoriamente uniforme entre 0,056 mm e 1,064 mm de forma que a razão $d_{\max}/d_{\min}$ seja aproximadamente 2 e são representativos de um arenito turbidítico em um campo produtor de petróleo. Com estas dimensões o número de esferas geradas foi de 38.392 e as diferentes faces da amostra podem ser vistas na Figura 3.1.

Tabela 3.1: Sugestões de preparação de amostra conforme a ISRM [51] e a ASTM [52] para ensaios de compressão. Os parâmetros H, D, $d_{\max}$ e n se referem ao comprimento, diâmetro da amostra, diâmetro máximo do grão e número de amostras requeridas, respectivamente. A ASTM não determina o número de amostras explicitamente, mas sugere o valor 5.

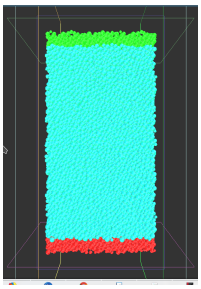
ISRM [51]	H/D	$= 2$ - 3
	D	≥ 50 mm
	$D/d_{\max}$	≥ 20
	n	≥ 5
ASTM [52]	H/D	$= 2$ - $2{,}5$
	D	≥ 47 mm
	$D/d_{\max}$	≥ 10



3.1(a): Vista da face paralela ao plano xy.



3.1(b): Vista da face paralela ao plano yz.



3.1(c): Vista da face paralela ao plano zx.

Figura 3.1: Amostra utilizada nos ensaios triaxiais. O centro e raio das esferas foram geradas aleatoriamente. As cores vermelho e verde das extremidades indicam os locais onde serão aplicadas as tensões σ_1.

Após a definição das dimensões da amostra e das esferas, os ensaios serão simulados. A fim de otimizar o tempo de processamento, foram realizados ensaios triaxiais com três tensões confinantes distintas, considerando em todos os casos $\sigma_2 = \sigma_3$: 1 MPa, 5 MPa e 10 MPa, o que resulta em um tempo de processamento médio de 4h 30 min!

Para se calcular as curvas $\sigma_d \times \epsilon_a$ e $\epsilon_V \times \epsilon_a$, que foram chamadas de experimentais, atribuíram-se certos valores para seus microparâmetros. Assumiu-se o conjunto da Tabela 3.2, que foram os obtidos via tentativa e erro em modelagem do arenito Fontainebleau [8, 9].

Tabela 3.2: Microparâmetros para o arenito Fontainebleau conforme referências [8, 9].

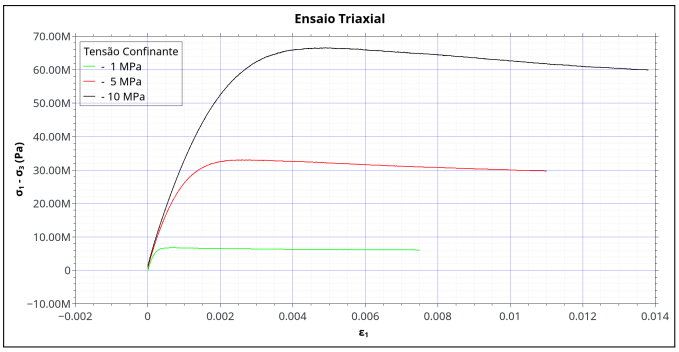
ρ (kg/m^3)	γ_{int} -	E_m (GPa)	k_s/k_n -	t (MPa)	c (MPa)	ϕ_b (°)	ϕ_c (°)
2700	1,5	50	0,35	4,5	45	18	18

Este conjunto de microparâmetros resultam nas curvas da Figura 3.2. Os seus macroparâmetros estão listados na Tabela 3.3.

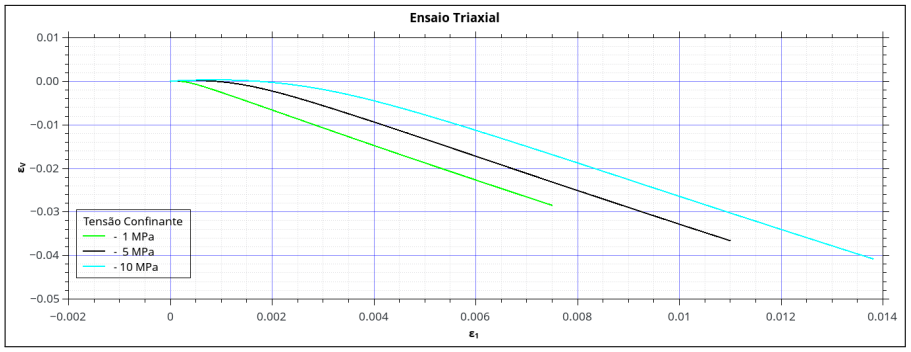
Tabela 3.3: Macroparâmetros obtidos dos ensaios do arenito Fontainebleau com os dados da Tabela 3.2.

E (GPa)	ν -	ϕ (°)	C_0 (MPa)
24,7 ± 0,1	0,25 ± 0,02	50,245 ± 0,002	0 ± 0,3

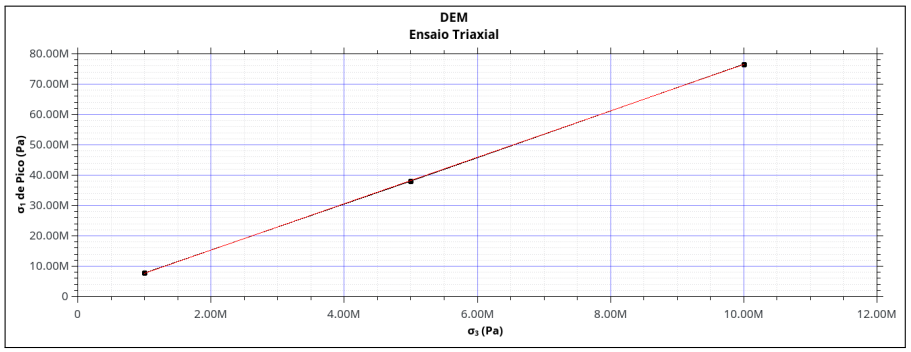
PUC-Rio - Certificação Digital Nº 1712747/CA



3.2(a): Gráficos $\sigma_d \times \epsilon_a$.



3.2(b): Gráficos $\epsilon_V \times \epsilon_a$.



3.2(c): Gráfico $\sigma_1 \times \sigma_3$ de pico.

Figura 3.2: Cruvas do ensaio triaxial do arenito Fontainebleau com tensões confinantes de 1 MPa, 5 MPa e 10 MPa, calculado com os microparâmetros da Tabela 3.2. Estas curvas serão chamadas de experimentais.

PUC-Rio - Certificação Digital Nº 1712747/CA

3.2.2
Densidade (ρ)

A seguir realizou-se uma análise de sensibilidade, avaliando como cada variável influencia no cálculo de R_2. Iniciando-se com a densidade do material, foi possível perceber que há baixíssima influência desta variável, conforme visto na Tabela 3.4, e desta maneira foi mantido o valor constante de 2700 kg/m^3 em todas as simulações. Este resultado já era esperado uma vez que numa simulação quase estática, em que a aceleração da gravidade é irrelevante, a densidade do material não influencia nos ensaios.

Tabela 3.4: Valores de R_2 em função da densidade da amostra. A função objetivo R_2 foi calculada comparando-se as curvas calculadas com as experimentais cuja a densidade foi de 2700 kg/m^3.

Densidade (kg/m^3)	R_2 (adim)
2000	1,01 10^{-5}
2100	6,93 10^{-6}
2200	5,81 10^{-6}
2300	3,85 10^{-6}
2400	2,98 10^{-6}
2500	2,15 10^{-6}
2600	1,60 10^{-6}
2700	1,32 10^{-6}
2800	2,24 10^{-6}
2900	2,84 10^{-6}
3000	4,92 10^{-6}

PUC-Rio - Certificação Digital Nº 1712747/CA

3.2.3
Interaction Range (γ_{int})

A descrição clássica do DEM diz que somente os elementos que estão em contato são conectados, porém, mais recentemente foi proposta uma faixa de interação, dada pela variável γ_{int}. O valor deste parâmetro está intimamente ligado com os raios dos elementos. Assim, de acordo com Scholtès e Donzé[8], para uma relação $r_{\max}/r_{\min} = 2$, o valor de γ_{int} deve ser, no máximo, 1,5.

Na avaliação do comportamento da R_2 em relação à variação de γ_{int} percebeu-se um aumento exagerado no tempo computacional, chegando a mais de 24h de processamento para a simulação dos ensaios com $\gamma_{\text{int}} = 1$. Desta maneira, atribuiu-se em todas as simulações o valor $\gamma_{\text{int}} = 1{,}5$. Nesta geometria, o número de coordenação torna-se igual a 13,02, ou seja, cada elemento tem em média ligações com 13 vizinhos.

3.2.4
Velocidade de Carregamento (v_l)

Em um ensaio triaxial a velocidade de carregamento é uma variável bastante relevante. Deve ser baixa o suficiente para que se garanta à amostra um equilíbrio quase estático durante o ensaio. Este equilíbrio é bastante importante para que a amostra não experimente um aumento de resistência ou que o material tenha respostas inesperadas.

Conforme descrito por Fjaer et al.[53], em um ensaio real as velocidades de compressão devem ser da ordem 10^{-10} m/s a 10^{-4} m/s, de acordo com o tipo de rocha e o objetivo do ensaio. Em modelagens DEM alguns autores usam valores que variam de 0,016 m/s a 0,200 m/s [15, 54, 55, 56].

Uma vez que a lógica de cálculo envolve a solução das equações de movimento pela segunda lei de Newton, o passo de tempo deve ser pequeno o suficiente para a análise quase estática. Sabendo-se que o passo de tempo aplicado a este tipo de simulação é da ordem de 10^{-5} s, uma velocidade de 0,125 m/s, como a utilizada neste trabalho, implica em cerca de 100.000 ciclos de cálculos para a placa se mover de 1 mm, o que é bastante razoável para uma simulação. Utilizou-se este valor de velocidade de carregamento em todas as simulações.

3.2.5
O Fator de Amortecimento Numérico (λ_d)

O amortecimento numérico é um maneira artificial de se atenuar a energia cinética dos elementos com o objetivo de se reproduzir o comportamento geral do material modelado. Normalmente encontra-se na literatura valores que variam de 0,015, considerado muito baixo, a 0,7, muito alto [8, 54]. Com isto foram realizadas simulações com o objetivo de se avaliar o comportamento de R_2 conforme variam-se os valores de λ_d. O gráfico pode ser visto na Figura 3.3.

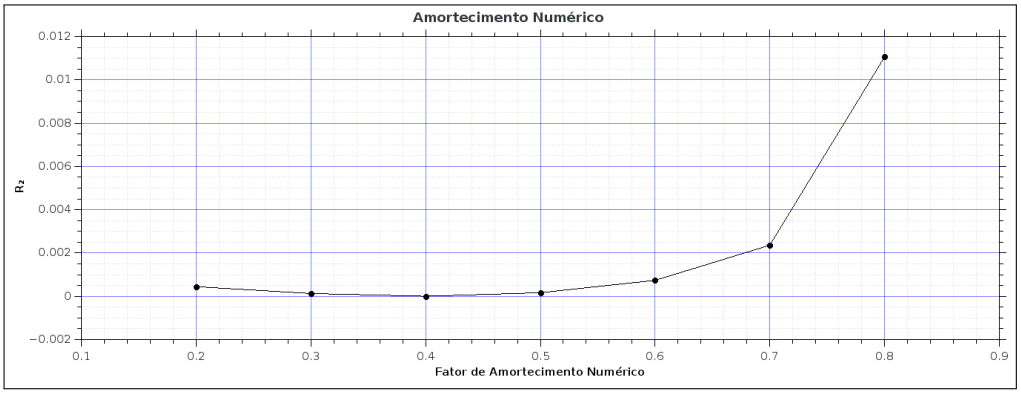


Figura 3.3: Gráfico de R_2 em função de λ_d.

PUC-Rio - Certificação Digital Nº 1712747/CA

Para a faixa de valores de λ_d normalmente visto na literatura, não há grande mudança no valor de R_2, mantendo-se abaixo de 0,0007. Exceto para a porção de valores $\lambda_d \geq 0{,}7$ pôde-se perceber um aumento mais acentuado de R_2. Desta maneira, manteve-se o valor constante $\lambda_d = 0{,}4$ em todos as simulações, conforme os cálculos apresentados na referência [8].

3.2.6 Microparâmetros

Após determinar as variáveis que serão utilizadas na otimização, o próximo passo é fazer uma análise de sensibilidade nos microparâmetros para avaliar qual a sua influência na função objetivo. Alguns dos microparâmetros foram limitados a valores normalmente encontrados na literatura [29, 31, 32], como por exemplo $k_s/k_n \leq 1$ e $\phi_b \leq 50°$. Quanto aos microparâmetros restantes observou-se que para $c > 55$ MPa não houve uma mudança considerável na resposta da função objetivo; em t >13,3 MPa ocorreu um erro simulação nas tensões de confinamento mais elevadas; para E_m abaixo de 5 GPa e acima de 120 GPa houve um aumento acentuado na resposta da função objetivo. Os resultados foram sintetizados na Tabela 3.5.

Sem nenhuma surpresa percebe-se que há diferentes níveis de sensibilidade para os diferentes microparâmetros. Dentro da faixa de variação permitida neste trabalho, o microparâmetro E_m é o mais significativo, passando por, em ordem de importância, ϕ_b e t (conforme gráficos da Figura 3.4). Por outro lado, há uma baixa sensibilidade nos microparâmetros c e k_s/k_n (Figura 3.5), que, por terem valores muito próximo do zero, não foram incluídos na otimização.

As variáveis ϕ_b e ϕ_c foram assumidas iguais em todas as simulações por um simples motivo: na versão do *software* Yade utilizada não havia sido implementado o modelo de contato JCF com estes dois microparâmetros independentes.

3.2.7 Análise de Tempo de Processamento

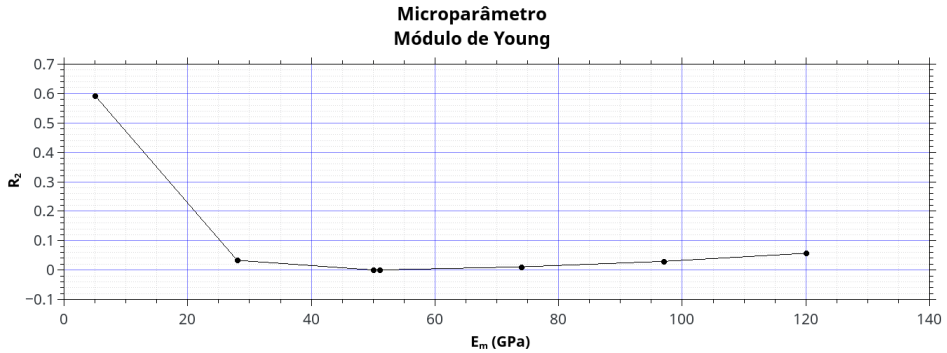
Outro ponto importante trata-se do tempo de processamento de cada chamada da função objetivo. A fim de tentar reduzir este tempo sem comprometer a qualidade dos resultados, realizou-se uma análise do tempo de processamento em função da quantidade de processadores utilizados. Os gráficos da Figura 3.6 mostram que o tempo mínimo é encontrado quando se utiliza nove processadores em cada ensaio independentemente.

PUC-Rio - Certificação Digital Nº 1712747/CA

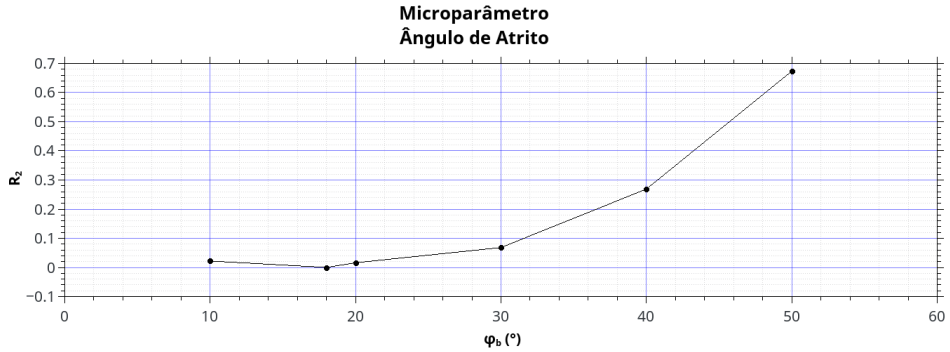
Tabela 3.5: Compilação do resultado da análise de sensibilidade de R_2 usando os microparâmetros como dados de entrada nas simulações. Exceto onde indicado, os microparâmetros foram mantidos fixos conforme os valores da Tabela 3.2.

Microparâmetro	Valor	R_2
c (MPa)	10,0	0,0004
	21,3	0,0001
	32,5	$< 10^{-4}$
	43,8	$< 10^{-4}$
	45,0	$< 10^{-4}$
	55,0	$< 10^{-4}$
$\phi_b = \phi_c$ (°)	10	0,0219
	18	0,0000
	20	0,0016
	30	0,0682
	40	0,2689
	50	0,6735
k_s/k_n	0,25	0,0001
	0,35	$< 10^{-4}$
	0,44	0,0001
	0,63	0,0004
	0,81	0,0007
	1,00	0,0010
t (MPa)	1,0	0,0002
	4,5	$< 10^{-4}$
	8,0	0,0020
	11,5	0,0302
	13,3	0,0888
E_m (GPa)	5	0,5932
	28	0,0328
	50	$< 10^{-4}$
	51	$< 10^{-4}$
	74	0,0105
	97	0,0292
	120	0,0597

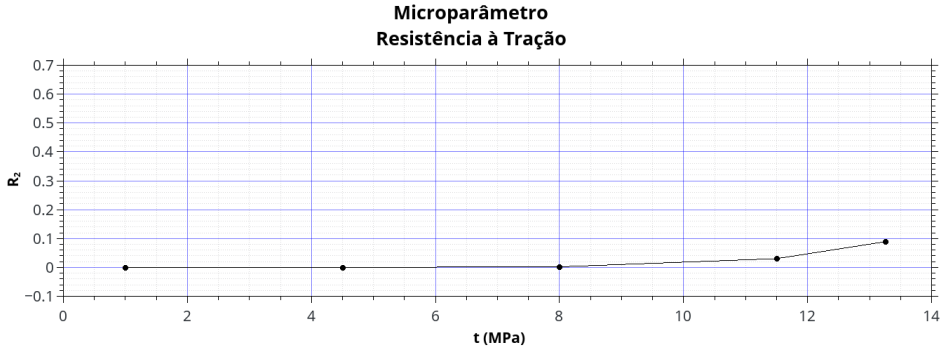
PUC-Rio - Certificação Digital Nº 1712747/CA



3.4(a): Gráfico $R_2 \times E_m$.



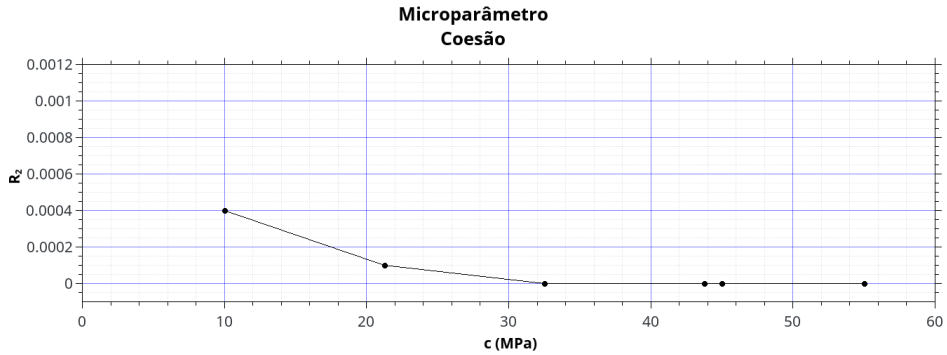
3.4(b): Gráfico $R_2 \times \phi_b$.



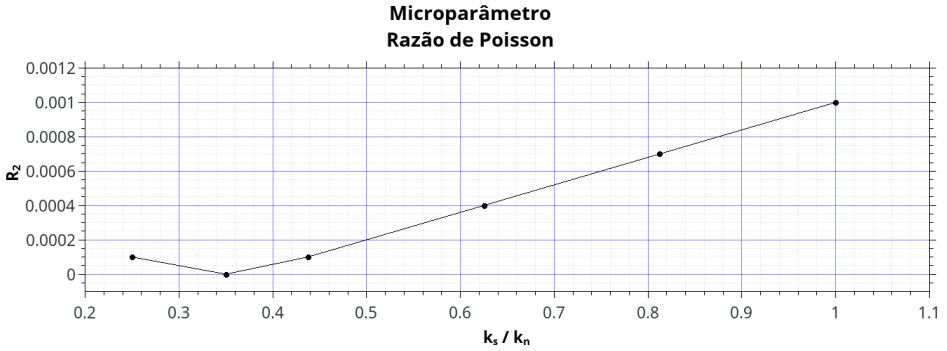
3.4(c): Gráfico $R_2 \times t$.

Figura 3.4: Análise de sensibilidade dos microparâmetros mais relevantes para a função objetivo: E_m, ϕ_b e t. A escala do eixo R_2 varia de -0,1 a 0,7.

PUC-Rio - Certificação Digital Nº 1712747/CA



3.5(a): Gráfico $R_2 \times c$.



3.5(b): Gráfico $R_2 \times k_s/k_n$.

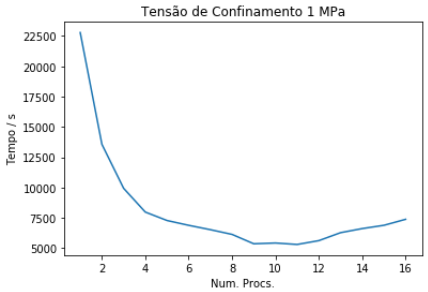
Figura 3.5: Análise de sensibilidade dos microparâmetros menos relevantes para a função objetivo: c e k_s/k_n. A escala do eixo R_2 varia de -0,0001 a 0,0012.

Com base neste resultado e, levando em consideração que a máquina mais robusta utilizada contava com 16 núcleos, a maneira mais rápida de efetuar o conjunto de simulação dos três ensaios foi utilizando-se cinco núcleos em cada ensaio de maneira a calculá-los todos ao mesmo tempo. Para as outras máquinas mais simples, assumiu-se que o tempo de processamento seria linearmente proporcional e procedeu-se da mesma forma. De uma maneira geral uma chamada da função objetivo leva em média 4h 30 min, podendo-se variar de aproximadamente 2h até 9h ininterruptas.

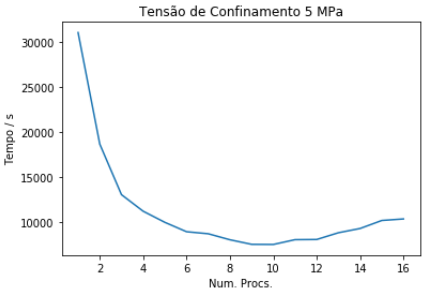
3.2.8 Indeterminismo Devido ao Multiprocessamento

A maneira como o Yade foi codificado permite que, ao se usar o multiprocessamento, sejam atribuídos porções de cálculos aleatoriamente a cada processador [21]. Isto implica num indeterminismo ao resultado final. Para avaliar a influência desta questão ao resultado, foram repetidas 21 vezes o mesmo ensaio, concluíndo-se que o resultado na função objetivo é menor que 10^{-5}, ou seja, não há influência significativa que comprometesse os resultados.

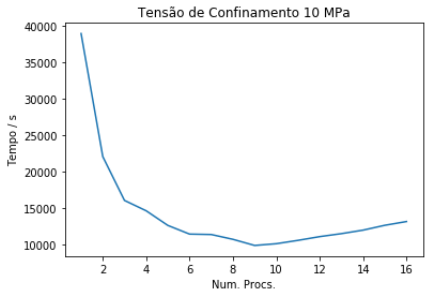
PUC-Rio - Certificação Digital Nº 1712747/CA



3.6(a): $\sigma_3 = 1$ MPa.



3.6(b): $\sigma_3 = 5$ MPa.



3.6(c): $\sigma_3 = 10$ MPa.

Figura 3.6: Tempo de processamento em função do número de processadores para a simulação do ensaio triaxial com tensão confinantes variadas. Estes testes foi realizados numa máquina equipada com processador Intel Xeon CPU E5-2687W 0 @ 3.10 GHz.

3.3 *Generalized Simulated Annealing*

Para se proceder com a busca global utilizando o GSA se faz necessária a escolha de seus parâmetros, tais como temperatura inicial e critérios de parada. Estas escolhas devem ser tais que a busca seja efetiva ao máximo possível, evitando-se assim o "armadilhamento" em mínimos locais.

O ponto de partida, ou ponto inicial, foi escolhido aleatoriamente dentre da faixa de valores permitidas para as variáveis e utilizado o mesmo em todas as otimizações.

3.3.1 Par (q_V; q_A)

É sabido que cada problema apresenta o par (q_V; q_A) em que a otimização se torna mais eficiente. De maneira geral quanto menor o valor de q_A, respeitando o limite de precisão numérica na avaliação da Equação 2-12, mais eficiente será a busca. Nas máquinas atuais este limite acontece quando $q_A = -5$, sendo este o valor atribuído. Para o parâmetro q_V, por experiência pessoal, os valores mais eficientes variavam entre 2,0 e 2,3. Isto possibilitou a escolha do par (q_V; q_A) = (2,2; -5,0).

PUC-Rio - Certificação Digital Nº 1712747/CA

3.3.2 Temperatura Inicial (T_0)

A escolha da temperatura inicial adequada está intimamente relacionada com a eficiência da otimização. Uma temperatura inicial muita alta faz com que sejam necessárias muitas cadeias de Markov para que haja a convergência. Por outro lado T_0 muito baixa tende a deixar a otimização presa num mínimo local. Esta escolha deve ser baixa o suficiente de forma a não precisar de muitas iterações para convergir e alta o suficiente para que haja a possibilidade de salto em todo seu domínio.

Para avaliar o valor de T_0 mais adequado, toma-se como base a variável que apresente a maior extensão ou maior domínio que, segundo a Tabela 3.5, é o microparâmetro E_m. Sendo assim, é preciso que haja um ajuste da temperatura inicial para que os sorteios aleatórios, conforme a distribuição de Tsallis (Equação 2-10), atinja os seus limites. O ajuste é feito com o microparâmetro que apresente a maior variação, pois é o limitante da temperatura inicial. Outras variáveis com menor domínio, serão contempladas automaticamente. Este é um artifício se para encontrar a temperatura inicial que melhor se ajusta ao problema proposto, de forma que não seja desnecessariamente alta, o que implicaria em um maior tempo de processamento para atingir a convergência.

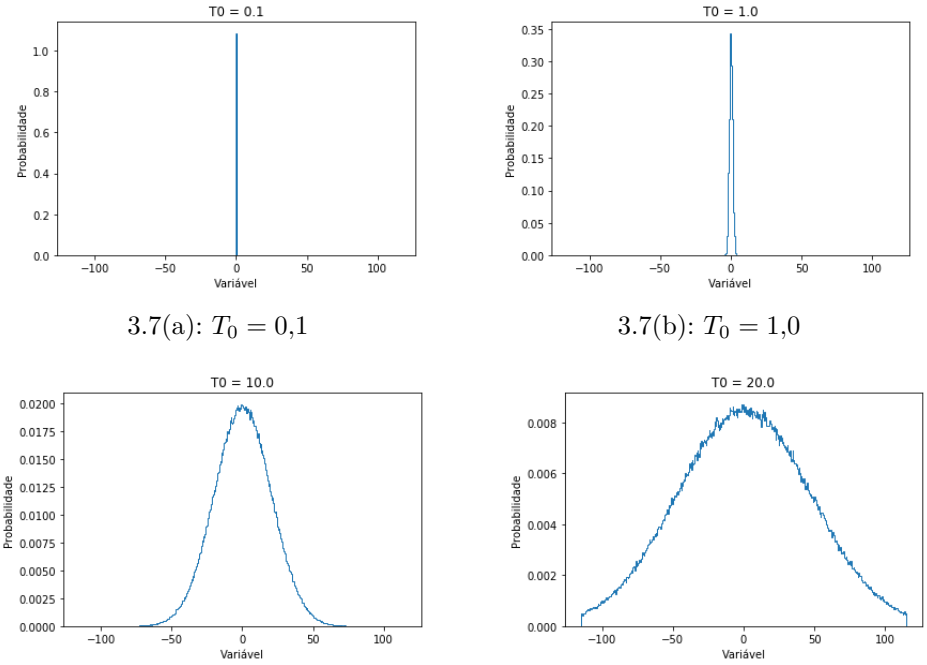
O microparâmetro E_m pode variar de 5 GPa a 120 GPa, ou seja, média 62,5 GPa e variação de ± 57,5 GPa. A temperatura inicial deve tal que garanta que os sorteios alcancem os limites de ± 57,5 a partir do valor médio. Desta maneira, testou-se a função distribuição para várias temperaturas de forma a atingir o critério descrito. Os histogramas podem ser vistos na Figura 3.7.

Os testes realizados com as temperaturas $T_0 = 0{,}1$ e $T_0 = 1{,}0$ apresentou sorteios muito próximos de centro da distribuição, não atingindo o critério discutido anteriormente. Já o caso em que $T_0 = 20$, uma parte considerável do histograma encontra-se fora dos limites. Isto indica que esta temperatura inicial já está alta para o problema. Assim a escolha mais apropriada torna-se $T_0 = 10$.

3.3.3 Critérios de Parada

Para se evitar que o código rode indefinidamente foi necessário determinar os critérios de parada. Neste caso estes critérios podem ser por convergência da função objetivo a um determinado limite mínimo ou por atingir um número máximo de cálculos.

PUC-Rio - Certificação Digital Nº 1712747/CA



3.7(a): $T_0 = 0{,}1$ 3.7(b): $T_0 = 1{,}0$

3.7(c): $T_0 = 10{,}0$ 3.7(d): $T_0 = 20{,}0$

Figura 3.7: Histograma do sorteio aleatório para várias temperaturas distintas. Pode-se perceber que a amplitude da distribuição aumenta à medida que se aumenta a temperatura.

3.3.3.1
Convegência da Função Objetivo

Como discutido na seção 3.2.6, os microparâmetros utilizado na otimização apresentam sensibilidades distintas para a função objetivo. Desta maneira considera-se a otimização convergida quando atingir a condição $R_2 \leq 0{,}0002$, pois isto dá um erro no microparâmetro E_m em média da ordem de 0,7 %.

O valor apresentado para o critério de convergência pela função objetivo é válido somente para este material, com estas curvas. Caso sejam acrescentados outras tensões de confinamento ou outros experimentos, deve-se realizar esta análise novamente para determinar o valor da função objetivo que indique a convergência desejada. Para o caso de dados reais, uma vez não se sabe quais são os seus microparâmetros, este critério de convergência perde o sentido. Deve-se, então, utilizar somente o critério discutido abaixo.

3.3.3.2
Número Máximo de Cadeias de Markov

Para os casos em que não fosse atingida a convergência pela função objetivo, foi necessária a limitação do máximo de pontos calculados. A temperatura final teve que ser em torno de 0,04 para que os sorteios fossem próximos o sufi-

PUC-Rio - Certificação Digital Nº 1712747/CA

ciente do mínimo global e que as variações na função objetivo fossem menores que 0,0002.

Solucionando a Equação 2-11 para t e substituindo o valor $q_V = 2{,}2$, o número máximo de cadeias de Markov torna-se aproximadamente 120. Felizmente a convergência aconteceu muito antes disso, pois caso contrário seria preciso esperar cerca de 135 dias de processamento!

3.4 Aspectos Computacionais

Os cálculos foram realizados utilizando a versão 2017.01a do Yade, sobre um sistema operacional Debian 9. Infelizmente esta versão do Yade ainda conta com o Python 2. Desta forma, como ponte entre o Yade e o GSA, foi utilizada a versão 3.5 do Python com as bibliotecas NumPy 1.17.2 e SciPy 1.3.1.

No aspecto de *hardware* contou-se com três máquinas: Um Intel Xeon CPU E5-2687W 0 @ 3.10 GHz com 8 núcleos, 16 threads e 16 GB de RAM em máquina virtual; um Intel i7 3612QM @2.10 GHz 4 núcleos, 8 threads com 8 GB RAM e um Intel Core i7 6820HQ @ 2.70 GHz 4 núcleos com 8 GB de RAM em máquina virtual.

PUC-Rio - Certificação Digital Nº 1712747/CA

4
Resultados e Discussão

4.1
Dados Sintéticos

Foi possível realizar três otimizações independentes via GSA, atingindo a convergência com um total de 68 ± 47 simulações. Os microparâmetros encontrados podem ser vistos na Tabela 4.1.

Tabela 4.1: Microparâmetros otimizados via GSA.

Microparâmetro	1	2	3	Média	Erro
ϕ_b (°)	16,40	18,68	17,05	17 ± 1	5,5 %
t (MPa)	7,16	2,21	6,20	5 ± 3	11,1 %
E_m (GPa)	50,20	49,74	49,59	49,8 ± 0,3	0,4 %
$R_2 \cdot 10^{-5}$	11,03	2,28	3,77	5 ± 4	-

Como explicitado anteriormente, a sensibilidade em cada um dos microparâmetros é distinta e pode ser evidenciada a partir do erro relativo. Para o microparâmetro mais sensível, E_m, foi encontrado um erro bastante baixo de 0,4 %. Já para os microparâmetros t e ϕ_b foram encontrados os erros de 11,1 % e 5,5 %, respectivamente. Estes valores indicam que os microparâmetros puderam ser satisfatoriamente determinados com o GSA. Isto reduziria consideravelmente o esforço despendido do modelador.

A resposta macroscópica, que envolve E, ν, ϕ e C_0, pode ser vista na Tabela 4.2.

Tabela 4.2: Respostas macroscópicas do material simulado após a otimização.

Macro	1	2	3	Média	Erro
E (GPa)	24,1 ± 0,5	25,0 ± 0,1	24,1 ± 0,4	24,4 ± 0,4	1,2 %
ν	0,26 ± 0,02	0,25 ± 0,02	0,25 ± 0,02	0,25 ± 0,02	$< 10^{-3}$ %
ϕ (°)	63,29 ± 0,05	50,167 ± 0,002	50,090 ± 0,001	54,52 ± 0,03	8,5 %
C_0 (MPa)	0 ± 21	0 ± 233	0,2 ± 0,1	0 ± 135	$< 10^{-3}$ %

Em todos os macroparâmetros calculados os erros relativos obtidos foram menores que 10 %. Este é outro indicativo do quanto a otimização via GSA é uma técnica eficiente e precisa.

PUC-Rio - Certificação Digital Nº 1712747/CA

O GSA segue um caminho aleatório no hipervolume de variáveis. Desta maneira, cada vez que é realizada uma otimização, o caminho percorrido é completamente distinto. A evolução do valor mínimo da função objetivo pode ser visto no gráfico da Figura 4.1 e é percebido que a convergência ocorreu na primeira otimização após 20 reduções do valor da função objetivo, em 122 tentativas, enquanto as otimizações seguintes encontraram a convergência após cinco reduções (em 44 tentativas) e sete reduções (em 38 tentativas).

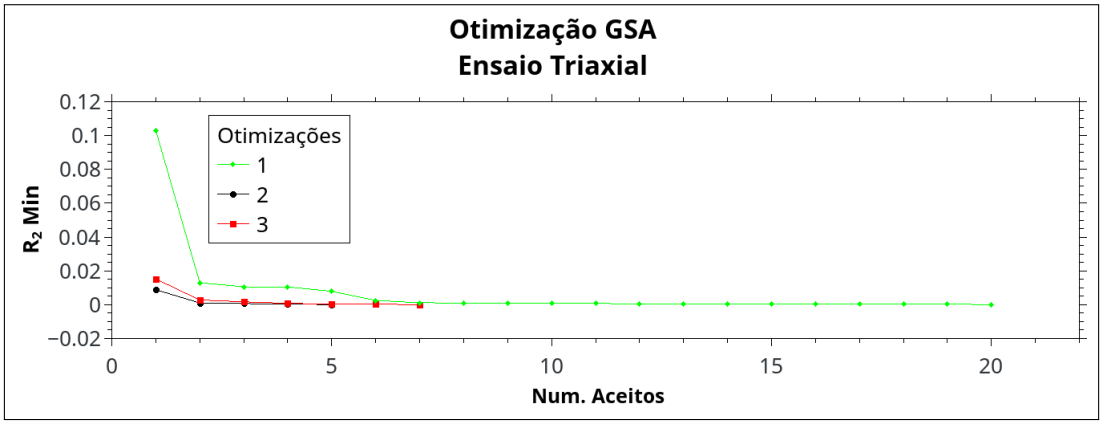
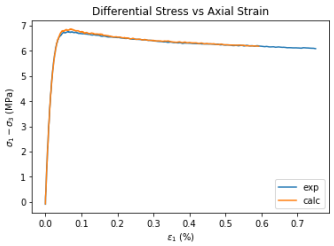


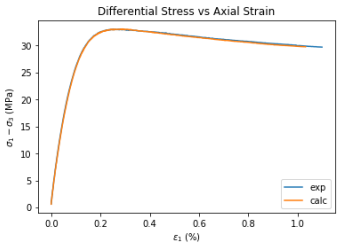
Figura 4.1: Evolução do valor mínimo da função objetivo nas três otimizações realizadas.

Não há uma correlação direta entre o número de reduções da função objetivo e número de tentativas, porém esta última representa o esforço computacional realizado. As tentativas que não são aceitas pelo critério de aceitação foram efetivamente calculadas, havendo um custo computacional envolvido na sua avaliação.

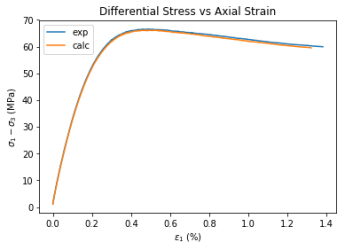
Algumas curvas otimizadas $\sigma_d \times \epsilon_a$ e $\epsilon_V \times \epsilon_a$ podem ser vistas nas Figuras 4.2 e 4.3.



4.2(a): $\sigma_3 = 1$ MPa.



4.2(b): $\sigma_3 = 5$ MPa.

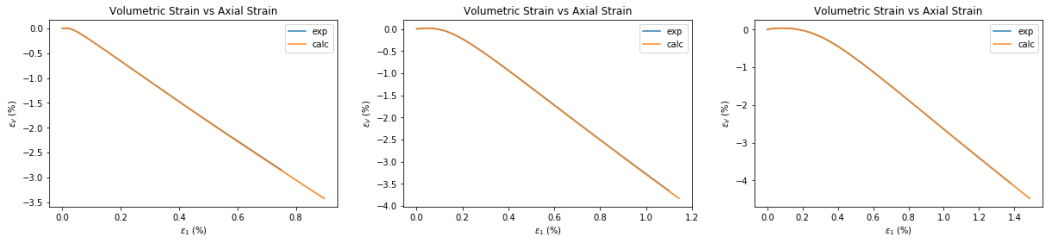


4.2(c): $\sigma_3 = 10$ MPa.

Figura 4.2: Comparação das curvas $\sigma_d \times \epsilon_a$ experimentais e otimizadas para as tensões confinantes de 1 MPa, 5 MPa e 10 MPa.

Este método de calibração é independente do modelo de contato, do tipo de simulação, quais ensaios se desejam simular e das curvas que representam o ensaio simulado. Isto significa que sua aplicação é bastante abrangente sem,

PUC-Rio - Certificação Digital Nº 1712747/CA



4.3(a): $\sigma_3 = 1$ MPa. 4.3(b): $\sigma_3 = 5$ MPa. 4.3(c): $\sigma_3 = 10$ MPa.

Figura 4.3: Comparação das curvas $\epsilon_V \times \epsilon_a$ experimentais e otimizadas para as tensões confinantes de 1 MPa, 5 MPa e 10 MPa.

a princípio, restrições à sua aplicação, podendo-se aumentar ou diminuir a quantidade de variáveis de acordo com o problema específico.

4.2 Dados Reais

Com o objetivo de aplicar esta técnica de calibração proposta em ensaios reais, utilizou-se o conjunto de dados de ensaio triaxial em amostras do Travertino Romano, publicados na dissertação de mestrado de Mauro G. Benedicto Junior [57]. Com o intuito de reduzir o tempo de processamento, usou-se somente os dados referentes aos ensaios de tensão confinante de 10 MPa, simulando o triaxial em uma amostra cúbica de lado 44,3 mm, com as esferas de raios aletoriamente uniforme entre 0,056 mm e 1,064 mm e a velocidade de carregamento de 0,2 m/s.

Realizaram-se independentemente três otimizações, partindo do mesmo ponto, com $q_V = 2{,}2$; $q_A = -5{,}0$ e $T_0 = 20$, seis variáveis e intervalo confome valores da Tabela 4.3. Encontrou-se o mínimo da função objetivo em $R_2 = 0{,}4 \pm 0{,}2$. A evolução do mínimo pode ser visto na Figura 4.4.

Tabela 4.3: Limites de cada variável na otimização dos dados do Travertino Romano.

Variável	Mínimo	Máximo
c (MPa)	100,0	500,0
ϕ_b (°)	10,0	75,0
k_s/k_n	0,25	0,70
t (MPa)	10,0	500,0
E_m (GPa)	1,0	500,0
λ_d	0,3	0,8

Para se atingir o mínimo em cada otimização precisou-se calcular uma média de 133,3 ± 42,6 simulações. As variáveis calibradas estão listadas na Tabela 4.4 e percebe-se que a terceira otimização resultou no valor de R_2 relativalmente alto em relação às otimizações anteriores.

PUC-Rio - Certificação Digital Nº 1712747/CA

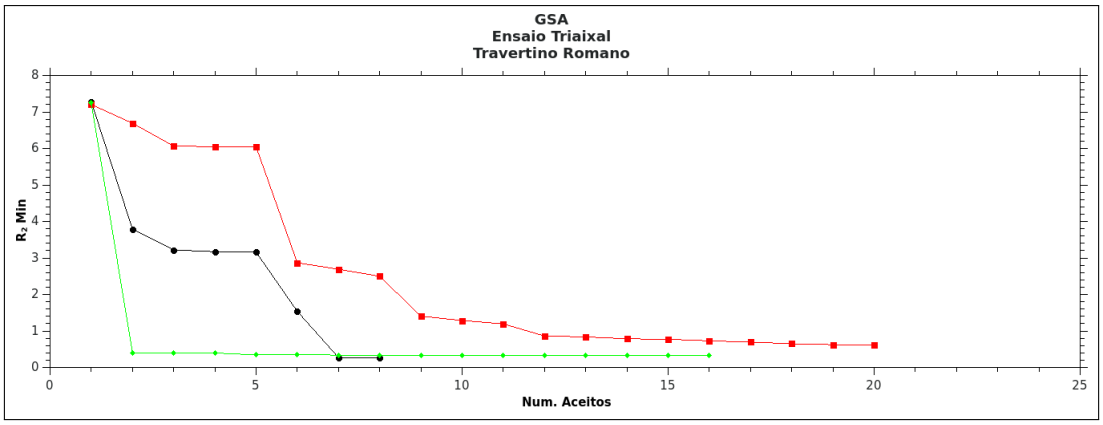


Figura 4.4: Evolução do valor mínimo da função objetivo nas três otimizações com dados reais do Travertino Romano. Cada otimização finalizou em $R_2 = 0{,}2577$ (preto), $R_2 = 0{,}3377$ (verde) e $R_2 = 0{,}6153$ (vermelho), resultando numa média de $R_2 = 0{,}4 \pm 0{,}2$.

Tabela 4.4: Variáveis otimizadas para o Travertino Romano via GSA.

	1	2	3	Valor Médio
c (MPa)	461,0	491,7	104,5	352,5 ± 215,2
ϕ_b (°)	12,2	47,7	72,9	44,3 ± 30,5
k_s/k_n	0,7	0,3	0,3	0,4 ± 0,2
t (MPa)	66,8	309,6	332,5	236,3 ± 147,2
E_m (GPa)	358,0	275,2	224,5	285,9 ± 67,4
λ_d	0,4	0,6	0,7	0,5 ± 0,1
R_2	0,2577	0,3377	0,6153	0,4 ± 0,2

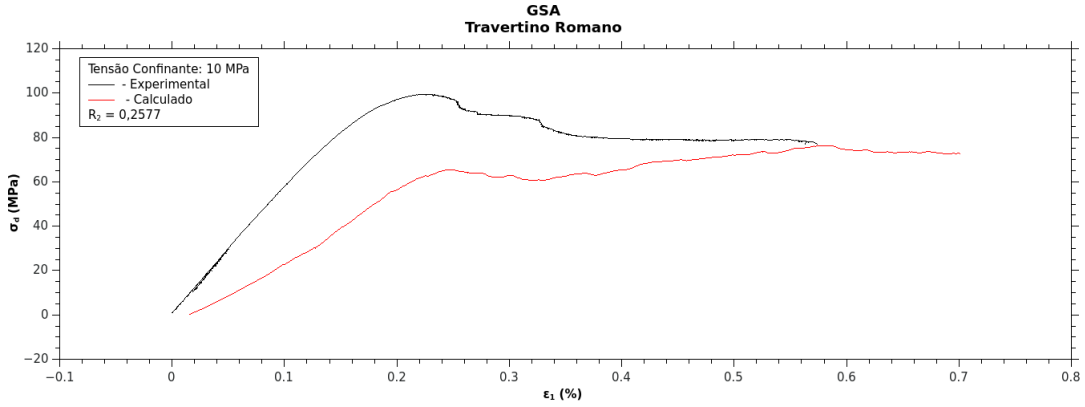
Os macroparâmetros após a otimização estão na Tabela 4.5 e as curvas de tensão-deformação axial e deformações volumétrica-axial estão na Figura 4.5.

Como dito anteriormente, o GSA visita aleatoriamente o hipervolume de variáveis e cada otimização é distinta da outra. A este fato inclui-se que, devido ao pouco tempo disponível ao final do projeto, não pôde-se realizar mais do que três otimizações. Este pequeno conjunto, apesar de não ser o suficiente para realizar uma estatística precisa, consegue indicar a facilidade e aplicabilidade do uso da otimização via GSA para calibração dos microparâmetros em simulações DEM.

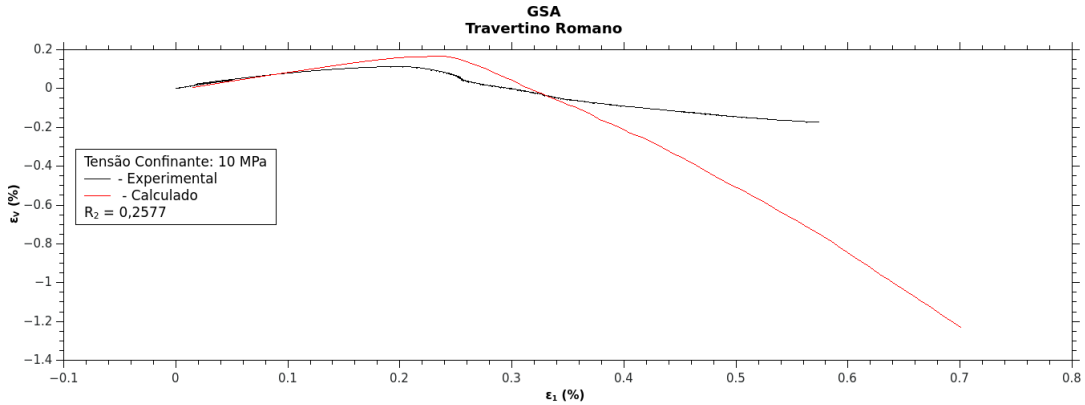
PUC-Rio - Certificação Digital Nº 1712747/CA

Tabela 4.5: Respostas macroscópica do material simulado após a otimização. Os valores foram obtidos a partir das curvas com o $R_2 = 0{,}2577$.

Macroparâmetro	Experimental	Otimizado	Erro
E (GPa)	54,8 ± 0,2	37,5 ± 0,4	31,6 %
ν	0,183 ± 0,002	0,0750 ± 0,0005	59 %



4.5(a): Gráficos de tensão-deformação axial experimental (preto) e otimizada (vermelho).



4.5(b): Gráficos de deformações volumétrica-axial experimental (preto) e otimizada (vermelho).

Figura 4.5: Comparação das curvas experimentais e calculadas após a otimização via GSA. Foram desenhadas as curvas otimizadas com o menor valor da função objetivo ($R_2 = 0{,}2577$).

PUC-Rio - Certificação Digital Nº 1712747/CA

5
Conclusões

Neste trabalho foi utilizado o *Generalized Simulated Annelealing* com o objetivo de se calibar os microparâmetros baseado na comparação entre as curvas experimentais e simuladas via DEM. Esta comparação foi realizada com o uso do funcional R_2 que leva em consideração a intensidade destas curvas, minimizando a área quadrática normalizada entre elas. Uma vez que a simulação dos experimentos têm como objetivo tentar reproduzir as suas curvas, a sua resposta macroscópica será automaticamente reproduzida.

Foram simulados os ensaios triaxiais com três tensões confinantes, considerando $\sigma_2 = \sigma_3$: 1 MPa, 5 MPa e 10 MPa. A partir daí foi realizada a avaliação das variáveis que influenciam nas curvas de tensão-deformação axial e deformações volumétrica-axial.

O modelo de contato utilizado, o JCF, é caracterizado por seis microparâmetros. Dentre eles somente três apresentaram sensibilidade signficativa: E_m, t e ϕ_b.

Foram realizadas três otimizações independentes, que convergiram para o mesmo ponto. Os erros relativos encontrados nos micro parâmetros foram de 5,5 % (ϕ_b), 11,1 % (t) e 0,4 % (E_m). Para os macroparâmentros encontrados, os erros foram 1,2 % (E), 8,5 % (ϕ) e menos de 10^{-3} % (ν e C_0).

No caso da otimização com dados reais, foram utilizados as curvas de tensão-deformação axial e deformações volumétrica-axial do Travertino Romano com tensão confinante de 10 MPa. Foram otimizadas seis variáveis e a função objetivo encontrada foi de $R_2 = 0{,}4 \pm 0{,}2$ o que resultou, nas respostas macroscópicas, em erros de 31,6 % para E e 59 % para ν.

Levando-se em conta o que foi exposto, é possível concluir que a automatização da calibração proposta neste trabalho elimina a necessidade da interação humana em cada passo, reduzindo a chance de subjetividade do modelador. Adicionalmente, devido às próprias características do GSA, este método pode ser aplicado para diferentes modelos de contato e diferentes conjuntos de simulações, a princícpio sem restrições, o que torna esta aplicação bastante ampla e interessante.

PUC-Rio - Certificação Digital Nº 1712747/CA

6
Sugestões Para Trabalhos Futuros

Todo trabalho envolve uma série de idéias que, à medida que são desenvolvidas, são levantadas novas questões. Neste trabalho não poderia ser diferente. Então, como sugestões para se avançar neste assunto:

1. Avaliar o uso de outros experimentos, por exemplo, ensaio de tração para uma caracterização geomecânica mais completa.

   Apesar de não haver indicativo de restrições, exceto computacional, quanto ao uso desta metodologia para calibração, não foi possível incluir outros tipos de ensaios além do triaxial com baixa tensão de confinamento;

2. Avaliar a aplicação das técnicas estatísticas de *Principal Component Analysis* e *Sphering* ou algum tipo de normalização.

   O objetivo seria o equilíbrio da sensibilidade de todas as variáveis na função objetivo;

3. Avaliar a influência dos parâmetros q_V e q_A na eficiência em se encontrar o mínimo global aplicado a este problema.

   Como dito anteriormente, cada problema apresenta o par q_V e q_A que a otimização se torna mais eficiente. Pode-se fazer uma avaliação nestes dois parâmetros para encontar o par mais eficiente para o problema de calibração em DEM;

4. Incluir na busca outros parâmetros discutidos na seção 3.2.

   Esta inclusão pode acarretar numa melhor reprodução das curvas experimentais, porém com o custo do aumento do tempo de cálculo em cada chamada da função objetivo;

5. Incluir outras geometrias para os elementos discretos.

   Com o mesmo intuito de se reproduzir mais fielmente as curvas experimentais;

PUC-Rio - Certificação Digital Nº 1712747/CA

7
Referências

1 ZIENKIEWICZ, O.; TAYLOR, R.; ZHU, J. *The Finite Element Method: Its Basis and Fundamentals*. Butterworth-Heinemann, 2013. (Butterworth-Heinemann, v. 1). ISBN 9781856176330. Disponível em: <https://books.google.com.br/books?id=nh_NmgEACAAJ>.

2 ATLURI, S. N.; ZHU, T. A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational Mechanics*, v. 22, n. 2, p. 117–127, Aug 1998. ISSN 1432-0924. Disponível em: <https://doi.org/10.1007/s004660050346>.

3 ATLURI, S. N.; ZHU, T. L. The meshless local Petrov-Galerkin (MLPG) approach for solving problems in elasto-statics. *Computational Mechanics*, v. 25, n. 2, p. 169–179, Mar 2000. ISSN 1432-0924. Disponível em: <https://doi.org/10.1007/s004660050467>.

4 ATLURI, S. N.; SHEN, S. The meshless local Petrov-Galerkin (MLPG) method: A simple & less-costly alternative to the finite element and boundary element methods. *Computer Modeling in Engineering & Sciences*, v. 3, n. 1, p. 11–52, 2002. Disponível em: <www.techscience.com/CMES/v3n1/24752>.

5 CUNDALL, P. A.; STRACK, O. D. L. A discrete numerical model for granular assemblies. *Géotechnique*, v. 29, n. 1, p. 47–65, 1979. Disponível em: <https://doi.org/10.1680/geot.1979.29.1.47>.

6 LEE, H.; MOON, T.; HAIMSON, B. C. Borehole breakouts induced in Arkosic sandstones and a descrete element analysis. *Rock Mechanics and Rock Engineering*, v. 49, n. 4, p. 1369–1388, Apr 2016. ISSN 1434-453X. Disponível em: <https://doi.org/10.1007/s00603-015-0812-0>.

7 CHAREYRE, B.; BRIANÇOM, L.; VILLARD, P. Theoretical versus experimental modeling of the achorage capacity of geotextitles in trenches. *Geosyntetics International*, v. 9, n. 2, p. 97–123, 2002. Disponível em: <https://doi.org/10.1680/gein.9.0212>.

8 SCHOLTÈS, L.; DONZÉ, F.-V. A DEM model for soft and hard rocks: Role of grain interlocking on strength. *Journal of the mechanics and Physics of solids*, v. 61, n. 2, p. 352–369, 2013. ISSN 0033-5096. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0022509612002268>.

9 SCHOLTÈS, L.; DONZÉ, F.-V. Modelling progressive failure in fractured rock masses unsing a 3D discrete element method. *International Journal of Rock Mechanics and Mining Sciences*, v. 52, p. 18–30, 2012. ISSN 1365-1609. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1365160912000391>.

PUC-Rio - Certificação Digital Nº 1712747/CA

10 POTYONDY, D.; CUNDALL, P. A bonded-particle model for rock. *International Journal of Rock Mechanics and Mining Sciences*, v. 41, n. 8, p. 1329–1364, 2004. ISSN 1365-1609. Rock Mechanics Results from the Underground Research Laboratory, Canada. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1365160904002874>.

11 LEE, S. J.; HASHASH, Y. M.; NEZAMI, E. G. Simulation of triaxial compression tests with polyhedral discrete elements. *Computers and Geotechnics*, v. 43, p. 92–100, 2012. ISSN 0266-352X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0266352X12000407>.

12 TSALLIS, C.; STARIOLO, D. A. Generalized simulated annealing. *Physica A: Statistical Mechanics and its Applications*, v. 233, n. 1, p. 395–406, 1996. ISSN 0378-4371. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0378437196002713>.

13 PINTO, T. C. *Inclusão do Método Generalized Simulated Annealing ao Código SATLEED*. 142 p. Dissertação (Mestrado em Física) — Instituto de Física, Universidade Federal da Bahia, Salvador, 2011. Disponível em: <www.superficie.ufba.br>.

14 CHAREYRE, B.; VILLARD, P. Dynamic spar elements and discrete element methods in two dimensions for the modeling of soil-inclusion problems. *Journal of Engineering Mechanics*, v. 131, n. 7, p. 689–698, 2005. Disponível em: <https://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9399%282005%29131%3A7%28689%29>.

15 CHO, N.; MARTIN, C.; SEGO, D. A clumped particle model for rock. *International Journal of Rock Mechanics and Mining Sciences*, v. 44, n. 7, p. 997–1010, 2007. ISSN 1365-1609. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1365160907000172>.

16 TONG, A.-T.; CATALANO, E.; CHAREYRE, B. Pore-scale flow simulations: Model predictions compared with experiments on bi-dispersed granular assemblies. *Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles*, v. 67, n. 5, p. 743–752, 2012. Disponível em: <https://doi.org/10.2516/ogst/2012032>.

17 CAMBORDE, F.; MARIOTTI, C.; DONZÉ, F. Numerical study of rock and concrete behaviour by discrete element modelling. *Computers and Geotechnics*, v. 27, n. 4, p. 225–247, 2000. ISSN 0266-352X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0266352X00000136>.

18 PODLOZHNYUK, A.; PIRKER, S.; KLOSS, C. Efficient implementation of superquadric particles in discrete element method within an open-source framework. *Computational Particle Mechanics*, v. 4, n. 1, p. 101–118, Jan 2017. ISSN 2196-4386. Disponível em: <https://doi.org/10.1007/s40571-016-0131-6>.

19 LU, G.; THIRD, J.; MüLLER, C. Discrete element models for non-spherical particle systems: From theoretical developments to applications. *Chemical Engineering Science*, v. 127, p. 425–465, 2015. ISSN 0009-2509. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0009250914007040>.

PUC-Rio - Certificação Digital Nº 1712747/CA

20 Itasca Consulting Group Inc. *PFC 5.0 Documentation*. [S.l.], 2015.

21 ŠMILAUER, V. et al. *Yade Documentation 2nd ed*. The Yade Project, 2015. Disponível em: <http://yade-dem.org/doc/>.

22 PENG, B. *Discrete Element Method (DEM) Contact Models Applied to Pavement Simulation*. 67 p. Dissertação (Mestrado em Ciências em Engenharia Civil) — Virginia Polytechnic Institute and State University, Blacksburg, 2014. Disponível em: <https://vtechworks.lib.vt.edu/bitstream/handle/10919/50399/Peng_B_T_2014.pdf>.

23 COETZEE, C. Review: Calibration of the discrete element method. *Powder Technology*, v. 310, p. 104–142, 2017. ISSN 0032-5910. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0032591017300268>.

24 NGUYEN, T.-T.; ANDRÉ, D.; HUGER, M. Analytic laws for direct calibration of discrete element modeling of brittle elastic media using cohesive beam model. *Computational Particle Mechanics*, v. 6, n. 3, p. 393–409, Jul 2019. ISSN 2196-4386. Disponível em: <https://doi.org/10.1007/s40571-018-00221-0>.

25 YOON, J. Application of experimental design and optimization to PFC model calibration in uniaxial compression simulation. *International Journal of Rock Mechanics and Mining Sciences*, v. 44, n. 6, p. 871–889, 2007. ISSN 1365-1609. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1365160907000135>.

26 DENG, Z. et al. Calibration of discrete element heat transfer parameters by central composite design. *Chinese Journal of Mechanical Engineering*, v. 30, n. 2, p. 419–427, Mar 2017. ISSN 2192-8258. Disponível em: <https://doi.org/10.1007/s10033-017-0072-x>.

27 CHEHREGHANI, S. et al. Bonded-particle model calibration using response surface methodology. *Particuology*, v. 32, p. 141–152, 2017. ISSN 1674-2001. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1674200117300226>.

28 RACKL, M.; HANLEY, K. J. A methodical calibration procedure for discrete element models. *Powder Technology*, v. 307, p. 73–83, 2017. ISSN 0032-5910. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0032591016308403>.

29 DO, H. Q.; ARAGÓN, A. M.; SCHOTT, D. L. Automated discrete element method calibration using genetic and optimization algorithms. *EPJ Web Conf.*, v. 140, p. 15011, 2017. Disponível em: <https://doi.org/10.1051/epjconf/201714015011>.

30 DO, H. Q.; ARAGÓN, A. M.; SCHOTT, D. L. A calibration framework for discrete element model parameters using genetic algorithms. *Advanced Powder Technology*, v. 29, n. 6, p. 1393–1403, 2018. ISSN 0921-8831. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0921883118300773>.

PUC-Rio - Certificação Digital Nº 1712747/CA

31 SIMONE, M. D.; SOUZA, L. M.; ROEHL, D. Estimating DEM microparameters for uniaxial compression simulation with genetic programming. *International Journal of Rock Mechanics and Mining Sciences*, v. 118, p. 33–41, 2019. ISSN 1365-1609. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1365160918307123>.

32 TAWADROUS, A. S. et al. Prediction of uniaxial compression PFDC3D model micro-properties using artificial neural networks. *International Journal for Numerical and Analytical Methods in Geomechanics*, Wiley Online Library, v. 33, n. 18, p. 1953–1962, 2009. ISSN 1096-9853. Disponível em: <https://doi.org/10.1002/nag.809>.

33 BENVENUTI, L.; KLOSS, C.; PIRKER, S. Identification of DEM simulation parameters by artificial neural networks and bulk experiments. *Powder Technology*, v. 291, p. 456–465, 2016. ISSN 0032-5910. Disponível em: <http://www.sciencedirect.com/science/article/pii/S003259101630002X>.

34 FAKHIMI, A.; VILLEGAS, T. Application of dimensional analysis in calibration of a discrete element model for rock deformation and fracture. *Rock Mechanics and Rock Engineering*, v. 40, n. 2, p. Article: 193, Jun 2006. ISSN 1434-453X. Disponível em: <https://doi.org/10.1007/s00603-006-0095-6>.

35 WANG, M.; CAO, P. Calibrating the micromechanical parameters of the PFC2D(3D) models using the improved simulated annealing algorithm. *Mathematical Problems in Engineering*, v. 2017, p. Article ID 6401835, 2017. Disponível em: <https://www.hindawi.com/journals/mpe/2017/6401835>.

36 HOLLAND, J. H. *Adaptation in Natural and artificial systems*: An introductory analysis with applications to biology, control, and artificial intelligence. [S.l.]: MIT Press, 1992. (Complex Adaptative Systems). ISBN 9780262082136.

37 JONES, D.; PERTTUNEN, C.; STUCKMAN, B. Lipschitzian optimization without Lipschitz constant. *Journal of Optimization Theory and Applications*, v. 79, n. 1, p. 157–181, Oct 1993. ISSN 1573-2878. Disponível em: <https://doi.org/10.1007/BF00941892>.

38 GABLONSKY, J.; KELLEY, C. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization*, v. 21, n. 1, p. 27–37, Sep 2001. ISSN 1573-2916. Disponível em: <https://doi.org/10.1023/A:1017930332101>.

39 KAELO, P.; ALI, M. M. Some variants of the controlled random search algorithm for global optimization. *Journal of Optimization Theory and Applications*, v. 130, n. 2, p. 253–264, Aug 2006. ISSN 1573-2878. Disponível em: <https://doi.org/10.1007/s10957-006-9101-0>.

40 RINNOOY KAN, A. H. G.; TIMMER, G. T. Stochastic global optimization methods part I: Clustering methods. *Mathematical Programming*, v. 39, n. 1, p. 27–56, Sep 1987. ISSN 1436-4646. Disponível em: <https://doi.org/10.1007/BF02592070>.

41 RINNOOY KAN, A. H. G.; TIMMER, G. T. Stochastic global optimization methods part II: Multi level methods. *Mathematical Programming*, v. 39, n. 1,

PUC-Rio - Certificação Digital Nº 1712747/CA

p. 57–78, Sep 1987. ISSN 1436-4646. Disponível em: <https://doi.org/10.1007/BF02592071>.

42 BONYADI, M. R.; MICHALEWICZ, Z. Particle swarm optimization for single objective continuous space problem: A review. *Evolutionary Computation*, v. 25, n. 1, p. 1–54, 2017. ISSN 1530-9304. Disponível em: <https://www.mitpressjournals.org/doi/10.1162/EVCO_r_00180>.

43 KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. *Science*, American Association for the Advancement of Science, v. 220, n. 4598, p. 671–680, 1983. ISSN 0036-8075. Disponível em: <http://science.sciencemag.org/content/220/4598/671>.

44 ČERNÝ, V. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, v. 45, n. 1, p. 41–51, Jan 1985. ISSN 1573-2878. Disponível em: <https://doi.org/10.1007/BF00940812>.

45 SZU, H.; HARTLEY, R. Fast simulated annealing. *Physics Letters A*, v. 122, n. 3, p. 157–162, 1987. ISSN 0375-9601. Disponível em: <http://www.sciencedirect.com/science/article/pii/0375960187907961>.

46 ANDRADE, M. D. de; MUNDIM, K. C.; MALBOUISSON, L. A. C. Convergence of the generalizes simulated annealing method with independent parametrs for the acceptance probability, visitation distribution and temperature functions. *International Journal of Quantum Chemistry*, v. 108, n. 13, p. 2392–2397, 2008. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/qua.21736>.

47 JOHNSON, S. G. *The NLopt nonlinear-optimization package*. 2020. Disponível em: <https://github.com/stevengj/nlopt>. Acesso em: 12.fev.2020.

48 NASCIMENTO, V. B. et al. The simulated annealing global search algorithm applied to the crystallography of surfaces by LEED. *Surface Review and Letters*, v. 06, n. 05, p. 651–661, 1999. Disponível em: <http://www.worldscientific.com/doi/abs/10.1142/S0218625X99000639>.

49 XIANG, Y.; SUN, D. Y.; GONG, X. G. Generalized simulated annealing studies on structures and properties of Ni_n (n = 2 - 55) clusters. *The Journal of Physical Chemistry A*, v. 104, n. 12, p. 2746–2751, 2000. Disponível em: <https://doi.org/10.1021/jp992923q>.

50 VAN HOVE, M. A.; WEINBERG, W. H.; CHAN, C.-M. Methods of surface crystallography by LEED: Reliability factors (R-Factors). In: _____. *Low-Energy Electron Diffraction: Experiment, Theory and Surface Structure Determination*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986. cap. 6, p. 237–250. ISBN 978-3-642-82721-1. Disponível em: <https://doi.org;10.1007/978-3-642-82721-1_6>.

51 DISCUSSION. *International Journal of Rock Mechanics and Mining Sciences*, v. 36, n. 3, p. 279–289, 1999. ISSN 1365-1609. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0148906299000066>.

PUC-Rio - Certificação Digital Nº 1712747/CA

52 ASTM International/D7012-14e1. *Standard Test Methods for Compressive Strength and Elastic Moduli of Intact Rock Core Specimens Under Varying States of Stress and Temperatures*. West Conshohocken, PA, 2014. 9 p. Disponível em: <https://doi.org/10.1520/D7012-14E01>.

53 FJAER, E. et al. Petroleum related rock mechanics. In: _____. 2. ed. Amsterdam: Elsevier, 2008. (Developments in Petroleum Science, v. 53), cap. 7, p. 258–259. ISBN 978-0-444-50260-5.

54 HAZZARD, J. F.; YOUNG, R. P.; MAXWELL, S. C. Micromechanical modeling of cracking and failure in brittle rocks. *Journal of Geophysical Research: Solid Earth*, v. 105, n. B7, p. 16683–16697, 2000. Disponível em: <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2000JB900085>.

55 DING, X. et al. Effect of model scale and particle size distribution on pfc3d simulation results. *Rock Mechanics and Rock Engineering*, v. 47, n. 6, p. 2139–2156, Nov 2014. ISSN 1434-453X. Disponível em: <https://doi.org/10.1007.s00603-013-0533-1>.

56 LI, K.; CHENG, Y.; FAN, X. Roles of model size and particle size distribution on macro-mechanical properties of lac du bonnet granite using flat-joint model. *Computers and Geotechnics*, v. 103, p. 43–60, 2018. ISSN 0266-352X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0266352X18301721>.

57 BENEDICTO JUNIOR, M. G. *Análise Geomecânica Anisotrópica do Travertino Romano (Quaternário)*. 188 p. Dissertação (Mestrado em Engenharia Civil) — Departamento de Engenharia Civil, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2018.

58 ELSGOLTS, L. Calculus of variations. In: _____. *Differential Equations and the Calculus of Variations*. Moscow: MIR Publishers, 1977. cap. 6–8, p. 291–388. Disponível em: <https://books.google.com.br/books?id=1kn_ewEACAAJ>.

PUC-Rio - Certificação Digital Nº 1712747/CA

A
Demonstração do Mínimo para o Funcional R_2

Funcionais são objetos matemáticos bastante interessantes e aparecem em vários campos da Física e Engenharia como forma de Leis de Conservação. São quantidades variáveis cujos valores são determinados pela escolha de uma ou mais funções e normalmente são escritas na forma de uma integral. Um excelente material sobre este assunto é o livro de Elsgolts[58] no qual esta demonstração foi baseada.

Dado o funcional

$$v\left[y(x)\right] = \int_{x_0}^{x_1} F\left(x, y, y'\right) \mathrm{d}x \tag{A-1}$$

os seus extremos são determinados a partir da solução da equação de Euler-Lagrange que tem a forma

$$\frac{\partial F}{\partial y} - \frac{\mathrm{d}}{\mathrm{d}x}\frac{\partial F}{\partial y'} = 0. \tag{A-2}$$

Reescrevendo a equação 2-14 de uma forma mais simples,

$$v\left[y(x)\right] = \int_{x_0}^{x_1} y^2 \mathrm{d}x, \tag{A-3}$$

com $y(x_0) = y_0$ e $y(x_1) = y_1$. A solução da equação de Euler-Lagrange aplicada a este funcional é

$$y(x) = 0. \tag{A-4}$$

A função que extremiza o funcional é $y(x) = 0$ e passa pelos pontos de contorno somente para $y_0 = 0$ e $y_1 = 0$. Se os pontos y_0 e y_1 são nulos, então a função $y(x) = 0$ evidentemente minimiza o funcional da equação A-3. O valor de $v = 0$ é somente para $y = 0$ e em qualquer outra função $v[y(x)] \geq 0$.

PUC-Rio - Certificação Digital Nº 1712747/CA

B
Código Fonte do Ensaio Triaxial

O código fonte original foi obtido nos exemplos do Yade [21] para simulação de materiais com baixa coesão. Precisou-se fazer modificações para o objetivo deste trabalho.

Pode-se utilizá-lo em modo iterativo, via linha de comando, ou em modo *batch*, inserindo como dado de entrada uma tabela com os valores das variáveis descritas nas linhas `readParamsFromTable`.

```
1
2  ''' Script para simulacao de ensaio triaxial.'''
3
4  from __future__ import print_function
5  from yade import pack
6
7  ################################################
8  ###   DEFINING VARIABLES AND MATERIALS   ###
9  ################################################
10
11 # The following 5 lines will be used later for batch execution
12 nRead = readParamsFromTable(
13     cohesion = 45e6,           # 10 a 55 MPa
14     finalFricDegree_b = 18.0,  # 10 a 50 graus
15     #finalFricDegree_c = 18.0, # 10 a 50 graus
16     poisson = 0.35,            # 0,25 a 0,7
17     sigmaT = 4.5e6,            # de 1 a 50 MPa - 1/10 a 1/5 do
    UCS
18     young = 50e9,              # 5 a 120 GPa
19     conf_mpa = 10,             # Confinamento em MPa
20     density = 2700,            # 2000 a 3000 kg/m3
21     damp = 0.4,                # 0.2 a 0.8
22     unknownOk = True)
23
24 from yade.params import table
25
26 # Tensao de confinamento (Pa) - Compressao negativa
27 conf_mpa = table.conf_mpa
28 confinamento = conf_mpa * (-1e6)
29
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
key = '_triax_' + str(conf_mpa) + 'MPa' # put you simulation's
    name here
targetPorosity = 0.01  # the porosity we want for the packing
compFricDegree = 89 # initial contact friction during the
    confining phase (will be decreased during the REFD
    compaction process)

# Segundo Fjaer, 2008, pagina 259, o strain rate deve ser entre
    10^-7 a 10^-1 /s. Normalmente e da ordem de 10^-2.
# Strain rate ou loading  rate. A relacao entre strain rate e
    velocidade e R = v(t) / L0
# Sao recomendados valores entre 0.016 a 0.2 m/s
# Com L0 = 88.622e-3, varia de 0.1805 a 2.2568 /s.
# velocidade de carregamento em m/s.
loading_velocity = -0.125 # negativo = compressao
damp = table.damp  # damping coefficient
gama_int = 1.5     # Interaction range. Pode variar de 1 a 1.5.
    Se for negativo, desabilita.
stabilityThreshold = 0.001  # we test unbalancedForce against
    this value in different loops (see below)
density = table.density

# Material
cohesion = table.cohesion
finalFricDegree_b = table.finalFricDegree_b
#finalFricDegree_c = table.finalFricDegree_c
poisson = table.poisson
sigmaT = table.sigmaT
young = table.young

# Dimensoes do paralelepipedo
base = 44.311e-3
altura = 2.0 * base
rparticle = 0.8e-3   # Raio da particula
sdparticle = 0.33333 # Percentagem do raio para estabelecer o
    Maximo e minimo. Isto e rparticle +/- sdparticle %
rate = loading_velocity / altura # Conversao para strain rate

# corners of the initial packing
mn, mx = Vector3(0, 0, 0), Vector3(altura, base, base)

## create materials for spheres and plates
O.materials.append(JCFpmMat(cohesion = cohesion,
                            density = density,
                            frictionAngle = radians(
    finalFricDegree_b),
                            poisson = poisson,
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
                              tensileStrength = sigmaT,
                              young = young,
                              label = 'spheres'))

O.materials.append(FrictMat(young = 1e12,
                              poisson = 0.5,
                              frictionAngle = 0,
                              density = 0,
                              label = 'walls'))

## create walls around the packing
walls = aabbWalls([mn, mx],
                  thickness = 0,
                  material = 'walls')
wallIds = O.bodies.append(walls)

## use a SpherePack object to generate a random loose particles
    packing
# Empacotamento das esferas em uma caixa.
pred = pack.inAlignedBox((0.0, 0.0, 0.0), (altura, base, base))
sp = SpherePack()

clumps = False #turn this true for the same example with clumps
num_spheres = 1000 # number of spheres. Somente para clumps.
if clumps:
    ## approximate mean rad of the futur dense packing for
    latter use
    volume = (mx[0] - mn[0]) * (mx[1] - mn[1]) * (mx[2] - mn
    [2])
    mean_rad = pow(0.09 * volume / num_spheres, 0.3333)
    ## define a unique clump type (we could have many, see
    clumpCloud documentation)
    c1 = pack.SpherePack([((-0.2 * mean_rad, 0, 0), 0.5 *
    mean_rad), ((0.2 * mean_rad, 0, 0), 0.5 * mean_rad)])
    ## generate positions and input them in the simulation
    sp.makeClumpCloud(mn, mx, [c1], periodic = False)
    sp.toSimulation(material = 'spheres')
    O.bodies.updateClumpProperties()   #get more accurate clump
    masses/volumes/inertia
else:
    sp = pack.randomDensePack(pred,
                              radius = rparticle,
                              rRelFuzz = sdparticle,
                              memoizeDb = 'triaxTestOnBox.
    sqlite',
                              returnSpherePack = True,
                              spheresInCell = 10000)
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
108     es = sp.toSimulation(material = 'spheres', color = (0, 1,
    1))
109
110 bot = [O.bodies[s] for s in es if O.bodies[s].state.pos[0] <
    rparticle * 7.5]
111 top = [O.bodies[s] for s in es if O.bodies[s].state.pos[0] >
    altura - rparticle * 7.5]
112
113 for s in bot:
114     s.shape.color = Vector3(1, 0, 0)
115 for s in top:
116     s.shape.color = Vector3(0, 1, 0)
117
118 # Numero de elementos
119 print("Num. Elementos: ", len(O.bodies) - 6)
120
121 ##############################
122 ###   DEFINING ENGINES   ###
123 ##############################
124
125 triax = TriaxialStressController(
126         ## TriaxialStressController will be used to control
    stress and strain. It controls particles size and plates
    positions.
127         ## this control of boundary conditions was used for
    instance in http://dx.doi.org/10.1016/j.ijengsci.2008.07.002
128         maxMultiplier = 1.0 + 4e7 / young, # spheres growing
    factor (fast growth)
129         finalMaxMultiplier = 1.0 + 4e6 / young, # spheres
    growing factor (slow growth)
130         thickness = 0,
131         ## switch stress/strain control using a bitmask. What
    is a bitmask, huh?!
132         ## Say x=1 if stress is controlled on x, else x=0 (
    strain is controlled). Same for for y and z. A melhor opcao
    e colocar em binario.
133         # A ordem e inversa (z,y,x)
134         # Strain - 0
135         # Stress - 1
136         stressMask = 0b111,
137         internalCompaction = True) # If true the confining
    pressure is generated by growing particles
138
139 newton = NewtonIntegrator(gravity = Vector3(0, 0, -9.81),
    damping = damp)
140
141 O.engines = [ForceResetter(),
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
142             InsertionSortCollider([Bo1_Sphere_Aabb(
    aabbEnlargeFactor = gama_int,
143                                                    label = "
    Saabb"),
144                                    Bo1_Box_Aabb()]),
145             InteractionLoop([Ig2_Sphere_Sphere_ScGeom(
    interactionDetectionFactor = gama_int,
146                                                      label =
    "SSgeom"),
147                              Ig2_Box_Sphere_ScGeom()],
148                             [Ip2_FrictMat_FrictMat_FrictPhys()
    ,
149                              Ip2_JCFpmMat_JCFpmMat_JCFpmPhys()
    ],
150                             [
    Law2_ScGeom_JCFpmPhys_JointedCohesiveFrictionalPM(
    recordCracks = False,
151
                    Key = "_rachaduras",
152
                    label = 'interactionLaw'),
153
    Law2_ScGeom_FrictPhys_CundallStrack()]),
154     ## We will use the global stiffness of each body to
    determine an optimal timestep (see https://yade-dem.org/w/
    images/1/1b/Chareyre&Villard2005_licensed.pdf)
155             GlobalStiffnessTimeStepper(active = 1,
156                                        timeStepUpdateInterval
    = 100,
157
    timestepSafetyCoefficient = 0.8),
158             VTKRecorder(recorders = ['spheres', 'boxes', 'intr
    ', 'coordNumber', 'velocity', 'stress', 'bstresses', 'jcfpm'
    , 'cracks'],
159                         Key = "_rachaduras",
160                         label = 'vtk',
161                         iterPeriod = 20,
162                         dead = True),
163             triax,
164             TriaxialStateRecorder(iterPeriod = 100,
165                                   file = '/tmp/yade/
    WallStresses' + key[0], dead = True),
166             newton]
167
168 # Habilita o tracking da energia.
169 O.trackEnergy = True
170 # Display spheres with 2 colors for seeing rotations better
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
171 Gl1_Sphere.stripes = True
172
173 # Display controller
174 #if nRead == 1:
175     #yade.qt.Controller(), yade.qt.View()
176
177 #UNCOMMENT THE FOLLOWING SECTIONS ONE BY ONE
178 #DEPENDING ON YOUR EDITOR, IT COULD BE DONE
179 #BY SELECTING THE CODE BLOCKS BETWEEN THE SUBTITLES
180 #AND PRESSING CTRL+SHIFT+D
181
182 if abs(confinamento) > 0.0:
183     print('########################################')
184     print('###   APPLYING CONFINING PRESSURE   ###')
185     print('########################################')
186
187     # Aplicando a tensao de confinamento nas tres direcoes
188     triax.goal1 = triax.goal2 = triax.goal3 = confinamento
189
190     contagem = 0
191     while 1:
192         O.run(1000, True)
193         # the global unbalanced force on dynamic bodies, thus
    excluding boundaries, which are not at equilibrium
194         unb = unbalancedForce()
195         desv_rel = abs((triax.meanStress - confinamento) /
    confinamento)
196         contagem += 1
197         print("Contagem: {}".format(contagem))
198         print('unbalanced force: {0:.4f}, mean stress: {1:.4f}'
    .format(unb, triax.meanStress / 1e6))
199         print("Desvio Relativo: {:.4f}".format(desv_rel))
200         print("s11: {0:.2f}, s22: {1:.2f}, s33: {2:.2f}".format
    (-(triax.stress(triax.wall_right_id)[0] + triax.stress(triax
    .wall_left_id)[0]) / 2.0 / 1e6, -(triax.stress(triax.
    wall_top_id)[1] + triax.stress(triax.wall_bottom_id)[1]) /
    2.0 / 1e6, -(triax.stress(triax.wall_front_id)[2] + triax.
    stress(triax.wall_back_id)[2]) / 2.0 / 1e6))
201         print(" *************************** ")
202         if unb < stabilityThreshold and desv_rel <
    stabilityThreshold:
203             break
204
205     #O.save('/tmp/confinedState' + key + '.yade.gz')
206     print("###      Isotropic state saved      ###")
207 #else:
208     #print('########################################')
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
209     #print('############   ESTABILIZACAO   #########')
210     #print('########################################')
211     ## Aplicando a tensao de confinamento nas tres direcoes
212     #triax.goal1 = triax.goal2 = triax.goal3 = -0.001e6
213     ## 200 rodadas para a estabilizacao
214     #O.run(1000, True)
215 
216 print("################################")
217 print("####   DEVIATORIC LOADING   ###")
218 print("################################")
219 
220 # We move to deviatoric loading, let us turn internal
        compaction off to keep particles sizes constant
221 triax.internalCompaction = False
222 
223 ## set stress control on y and z, we will impose strain rate on
        x.
224 # A ordem e inversa (z,y,x)
225 # Stress - 1
226 # Strain - 0
227 triax.stressMask = 0b110
228 ## we define the lateral stresses during the test, here the
        same 10kPa as for the initial confinement.
229 triax.goal2 = triax.goal3 = confinamento
230 ## now goal1 is the target strain rate
231 #triax.wall_left_activated = True
232 triax.goal1 = rate
233 
234 ##Save temporary state in live memory. This state will be
        reloaded from the interface with the "reload" button.
235 O.saveTmp()
236 
237 from yade import plot
238 
239 # Fase de estabilizacao...
240 # ESTABILIZACAO
241 # Roda somente para o ensaio UCS, 1200 vezes compactando, para
        estabilizar.
242 if conf_mpa == 0:
243     O.run(1200, True)
244 else:
245     print('########################################')
246     print('#####   APPLYING ESTABILIZACAO   #####')
247     print('########################################')
248     while 1:
249         O.run(100, True)
250         s11_est = -(triax.stress(triax.wall_right_id)[0] +
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
    triax.stress(triax.wall_left_id)[0]) / 2.0 + confinamento
251         print("{} MPa".format(s11_est / 1e6))
252         if s11_est / 1e6 > 0:
253             print(' ')
254             print(' ESTABILIZADO ')
255             print(' ')
256             break
257     # Ponto de controle
258     a = O.iter
259     print("Ponto de Controle: {}".format(a))
260 
261 ##########################################################
262 ####    Example of how to record and plot data     ###
263 ##########################################################
264 
265 ### a function saving variables
266 def history():
267     # Ha uma diferenca entre a deformacao logaritmica, entregue
    pela funcao triax.strain
268     # e a deformacao de engenheiro. A relacao entre elas e dada
    pela correcao abaixo.
269     e11 = (exp(-triax.strain[0]) - 1.0)
270     e22 = (exp(-triax.strain[1]) - 1.0)
271     e33 = (exp(-triax.strain[2]) - 1.0)
272     # Modifiquei a convencao de sinais aqui.
273     s11 = -(triax.stress(triax.wall_right_id)[0] + triax.stress
    (triax.wall_left_id)[0]) / 2.0 + confinamento
274     s22 = -(triax.stress(triax.wall_top_id)[1] + triax.stress(
    triax.wall_bottom_id)[1]) / 2.0 + confinamento
275     s33 = -(triax.stress(triax.wall_front_id)[2] + triax.stress
    (triax.wall_back_id)[2]) / 2.0 + confinamento
276     i = O.iter - a
277     Ek = O.energy['kinetic']
278     Eg = O.energy['gravWork']
279     Ed = O.energy['nonviscDamp']
280     #Eu = O.energy['elastPotential']
281     #Ep = O.energy['plastDissip']
282     plot.addData(e11 = e11,
283                  e22 = e22,
284                  e33 = e33,
285                  s11 = s11,
286                  s22 = s22,
287                  s33 = s33,
288                  i = i)
289     ### Funcao para salvar as curvas num arquivo.
290     with open("/tmp/curvas_" + str(conf_mpa) + "MPa.txt", "a")
    as arq:
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
291         arq.write(str(i) + "\t" +
292                   str(e11) + "\t" + str(e22) + "\t" + str(e33)
    + "\t" +
293                   str(s11) + "\t" + str(s22) + "\t" + str(s33)
    + "\n")
294
295 ##############################################
296 # Para a simulacao quando o corpo romper e a
297 # tensao atingir a metade seu do valor maximo.
298 ##############################################
299 def stopIfDamaged(maxEps = 10e-3):
300     extremum = max(abs(s) for s in plot.data['s11'])
301     s = abs(plot.data['s11'][-1])
302     e = abs(plot.data['e11'][-1])
303     print("Iteracao: ", O.iter - a)
304     print("s11: {0:.2f} MPa, s22: {1:.2f} MPa, s33: {2:.2f} MPa
    ".format(plot.data["s11"][-1] / 1e6, plot.data["s22"][-1] /
    1e6, plot.data["s33"][-1] / 1e6))
305     print("e11: {0:.2f} %, e22: {1:.2f} %, e33: {2:.2f}  %".
    format(plot.data["e11"][-1] * 100, plot.data["e22"][-1] *
    100, plot.data["e33"][-1] * 100))
306     if conf_mpa <= 0.5:
307         porcentagem = 0.5
308     else:
309         porcentagem = 0.9
310     if O.iter - a > 1000 and s < porcentagem * extremum or e *
    100 > 0.7:
311         print('  ')
312         print('################################')
313         print('####  SIMULATION FINISHED   ###')
314         print('################################')
315         print('  ')
316         print('Ruptura a {:.4f} MPa'.format(extremum / 1e6))
317         print('Deformacao {:.4f} %'.format(e * 100))
318         print('Num. Iteracoes:  ', O.iter - a)
319         print('   ')
320         import sys
321         sys.exit(0)
322         O.pause()
323     return
324
325 ## include a periodic engine calling that function in the
    simulation loop
326 O.engines = O.engines + [PyRunner(iterPeriod = 20, command = '
    history()', label = 'recorder'), PyRunner(iterPeriod = 50,
    command = 'stopIfDamaged()', label = "teste_fraturado")]
327
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
328 dadoslst = ["i", "e11", "e22", "e33", "s11", "s22", "s33"]
329
330 with open("/tmp/curvas_" + str(conf_mpa) + "MPa.txt", "w") as
    arq:
331     arq.write("\t".join(dadoslst) + "\n")
332
333 # Numero de coordenacao
334 print("Numero de Coordenacao: {:.2f}".format(utils.
    avgNumInteractions(cutoff = 0.0, skipFree = False,
    considerClumps = False)))
335
336 O.run()
337 waitIfBatch()
```

PUC-Rio - Certificação Digital Nº 1712747/CA

C
Código Fonte do Acoplamento GSA-Yade

Por conta da API da versão do Yade utilizada neste trabalho ainda ser o Python 2, foi preciso trabalhar com no esquema escrita-leitura de arquivos externos para fazer a ponte entre o Yade e o GSA.

O código faz a leitura do arquivo com as informações da simulação gerados pelo Yade, realiza o cálculo da função R_2 e a otimização via GSA. Os novos valores das variáveis são escritas numa tabela que é lido pelo Yade, repetindo até algum critério de parada.

Utiliza-se, prioritariamente, a bilbioteca SciPy onde se encontra implementação bem interessante do GSA chamada *dual annealing*.

PUC-Rio - Certificação Digital Nº 1712747/CA

```
1
2  #!/usr/bin/env python3
3  # -*- coding: utf-8 -*-
4  """
5  Created on Thu Sep 12 21:22:54 2019
6
7  @author: felipe
8
9  Juncao do GSA com o Yade, com funcao objetivo o fator R2.
10 """
11 import os
12 import pandas as pd
13 import time
14 import matplotlib.pyplot as plt
15
16 from numpy import linspace
17 from scipy.optimize import dual_annealing as gsa
18 from scipy import integrate, interpolate
19 #
    ********************************************************************************

20 def rodar(variaveis):
21     ''' Funcao para escrever o arquivo "tabela" com as
    variaveis,
22     apagar os arquivos que nao sao importantes, tais como "
    ensaio_triaxial.py.*.png"
```

```
23      e "ensaio_triaxial.py.*.log", rodar o yade-batch e chamar a
        comparacao de funcoes.
24      '''
25      # Converte para MPa (e6) e GPa (e9).
26      variaveis = [variaveis[0],        # phi
27                   variaveis[1] * 1e6, # t
28                   variaveis[2] * 1e9] # Em
29
30      # Tensoes de confinamento
31      ensaios = [1, 5, 10]
32
33      # Escreve o arquivo com as variaveis.
34      with open("tabela", "w") as file:
35          file.write("finalFricDegree\tsigmaT\tyoung\tconf_mpa\n"
    )
36          for i in ensaios:
37              file.write("\t".join(map(str, variaveis)) + "\t" +
    str(i) + "\n")
38
39      # Pausa de 1 s para garantir a escrita do arquivo.
40      os.system("sleep 1")
41
42      global numtotal
43      # Print do progresso
44      print("Progresso: {0} de {1} ({2:.2f} %)".format(numtotal +
        1,
45            2 * len(variaveis) * 120 + 1,
46            100 * numtotal / (2 * len(variaveis) * 120 + 1)))
47
48      # Rodar o yade-batch.
49      # Atentar que o job-threads tem que ser:
50      # debian9 (4) = 2
51      # delorean (8) = 2
52      # debian (16) = 5
53      t1 = time.time()
54      os.system("yade-batch --job-threads=5 tabela
    ensaio_triaxial.py")
55      t2 = time.time()
56
57      # Mover os logs para o diretorio LOGS
58      os.system("mv *.log LOGS")
59
60      # Numero da iteracao
61      numtotal = numtotal + 1
62
63      print("Tempo de Processamento: {}".format(time.strftime("%H
    :%M:%S", time.gmtime(t2 - t1))))
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
64
65      # Leitura dos arquivos contendo as curvas.
66      # Curvas experimentais
67      dirtmp = "/tmp/"
68      curvexp = ["curvasbase_1MPa.txt", "curvasbase_5MPa.txt", "
        curvasbase_10MPa.txt"]
69      # Curvas calculadas
70      curvcalc = ["curvas_1MPa.txt", "curvas_5MPa.txt", "
        curvas_10MPa.txt"]
71
72      r2global = []
73
74      # Le as curvas (exp e calc) e chama o metodo do fator R2.
75      for e, c in zip(curvexp, curvcalc):
76          nomes = [dirtmp + e,
77                   dirtmp + c]
78
79          # Criacao do dataframe
80          data = dict(exp = pd.read_table(nomes[0], sep = "\t"),
81                      calc = pd.read_table(nomes[1], sep = "\t"))
82
83          # Calcula a deformacao volumetrica
84          for i in data.keys():
85              data[i]["ev"] = data[i]["e11"] + data[i]["e22"] +
        data[i]["e33"]
86
87          r2global.append(R2(data))
88
89      print("R2: {:.4f}".format(sum(r2global) / len(r2global)))
90      return (sum(r2global) / len(r2global))
91
92  #
        ********************************************************************************
93  def R2(f):
94      ''' Calculo do fator R2 de LEED aplicado ao problema de
        tensao-deformacao.
95      '''
96      # Primeira Etapa: Reamostra e faz a interpolacao das
        funcoes.
97      # Comparar o eixo x das duas funcoes
98      prim = max(min(f["exp"]["e11"]), min(f["calc"]["e11"]))
99      ultm = min(max(f["exp"]["e11"]), max(f["calc"]["e11"]))
100
101     # Refaz o eixo x com os limites definidos acima.
102     # O numero de amostras e o dobro do maximo de amostras
        entre a curva calculada e
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
103     # experimental.
104     x = linspace(prim, ultm, 2 * max(len(f["exp"]["e11"]), len(
    f["calc"]["e11"])))
105 
106     fatorr = 0
107     for curva in ["s11", "ev"]:
108         # Criacao das funcoes de interpolacao
109         expint = interpolate.interp1d(f["exp"]["e11"], f["exp"
    ][curva])
110         calcint = interpolate.interp1d(f["calc"]["e11"], f["
    calc"][curva])
111 
112         # Segunda Etapa: Calcula o fator R2.
113         # Calcula as integrais de das curvas exp e calc.
114 #         Ie = integrate.trapz(expint(x), x = x)
115 #         It = integrate.trapz(teoint(x), x = x)
116         # Constante de normalizacao de intensidade relativa
117 #         c = Ie / It
118         c = 1.0
119         # Constante de normalizacao
120         A2 = 1.0 / integrate.trapz(expint(x) ** 2, x = x)
121         # Funcao R2
122         r = A2 * integrate.trapz((expint(x) - c * calcint(x))
    ** 2, x = x)
123         # Funcao R2 ponderada por curva.
124         fatorr += 0.5 * r
125 #         print("Curva {0}: {1:.4f}".format(curva,r))
126 #     print("    ")
127 #     print("R2: {:.4f}".format(fatorr))
128 #     print(" ****** ")
129     return fatorr
130 
131 #
    ********************************************************************************

132 def callbackGSA(coord, e, context = 0):
133     ''' Para acompanhamento do progresso.'''
134     # Criterio de convergencia
135     cc = 0.0002
136 
137     print("    ")
138     print("Ponto aceito")
139     print("Ponto:", coord)
140     print("R2: {:.4f}".format(e))
141     print("    ")
142     # Escrita dos pontos aceitos num arquivo
143     with open("aceitos.txt", "a") as arquivo:
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
144         arquivo.write(str(e) + "\n")
145 
146     # Criterio de convergencia por limite minimo da funcao objetivo.
147     if e <= cc:
148         # Leitura dos arquivos contendo as curvas.
149         # Curvas experimentais
150         dirtmp = "/tmp/"
151         curvexp = ["curvasbase_1MPa.txt", "curvasbase_5MPa.txt", "curvasbase_10MPa.txt"]
152         # Curvas calculadas
153         curvcalc = ["curvas_1MPa.txt", "curvas_5MPa.txt", "curvas_10MPa.txt"]
154 
155         for e, c in zip(curvexp, curvcalc):
156             nomes = [dirtmp + e,
157                      dirtmp + c]
158 
159             # Criacao do dataframe
160             data = dict(exp = pd.read_table(nomes[0], sep = "\t"),
161                         calc = pd.read_table(nomes[1], sep = "\t"))
162 
163             # Plot das curvas
164             for i in data.keys():
165                 # Calcula a deformacao volumetrica
166                 data[i]["ev"] = data[i]["e11"] + data[i]["e22"] + data[i]["e33"]
167                 # Tensao Axial x Deformacao axial
168                 plt.plot(data[i]["e11"] * 100, data[i]["s11"] / 1e6, label = i)
169             plt.title("Differential Stress vs Axial Strain")
170             plt.xlabel("$\epsilon_{1}$ (%)")
171             plt.ylabel("$\sigma_{1} - \sigma_{3}$ (MPa)")
172             plt.legend()
173             plt.show()
174 
175             for i in data.keys():
176                 plt.plot(data[i]["e11"] * 100, data[i]["ev"] * 100, label = i)
177             plt.title("Volumetric Strain vs Axial Strain")
178             plt.xlabel("$\epsilon_{1}$ (%)")
179             plt.ylabel("$\epsilon_{V}$ (%)")
180             plt.legend()
181             plt.show()
182 
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
183         # Copia a solucao para o diretorio atual.
184         for j in curvcalc:
185             os.system("cp /tmp/" + j + " CURVAS/")
186 
187         return True
188 #
    ***********************************************************************


189 
190 #
    ###########################################################################


191 ############################## Programa principal
    ################################
192 #
    ###########################################################################


193 
194 # Inicio do processamento
195 t0 = time.time()
196 
197 # Apagar os arquivo irrelevantes
198 os.system("rm -rf tabela *.log LOGS aceitos.txt")
199 # Criar diretorio de logs e CURVAS
200 os.system("mkdir LOGS CURVAS")
201 
202 # Curvas experimentais
203 direxp = "/home/felipe/MESTRADO/Programas/Yade+GSA/"
204 curvexp = ["curvasbase_1MPa.txt", "curvasbase_5MPa.txt", "
    curvasbase_10MPa.txt"]
205 
206 # Copiar as curvas experimentais para a memoria
207 for i in curvexp:
208     os.system("cp " + i + " /tmp/")
209 
210 # Global
211 numtotal = 0
212 
213 # Ponto inicial [phi, t, Em]
214 X0 = [31.4, 9.0, 38.9]
215 # A solucao e:
216 # C   = 45
217 # phi = 18
218 # nu  = 0.35
219 # t   = 4.5
220 # Em  = 50
221 
```

PUC-Rio - Certificação Digital Nº 1712747/CA

```
222 # Parametros do GSA.
223 lw = [10.0,  1.0,   5.0]
224 up = [50.0, 15.0, 120.0]
225 T0 = 10.0
226 qv = 2.2
227 qa = -5.0
228 # Quantidade de cadeias de Markov.
229 maxiter = 120  # Nesta implementacao a cadeia tem tamanho 2*D.
230
231 ret = gsa(rodar, bounds = list(zip(lw, up)), maxiter = maxiter,
232           initial_temp = T0, visit = qv, accept = qa, seed =
    419303,
233           no_local_search = True, callback = callbackGSA, x0 =
    X0)
234
235 # Final do processamento
236 tf = time.time()
237
238 print("   ")
239 print(" ************************ ")
240 print("Ponto final: ")
241 #print("C   = {:.2f} MPa".format(ret.x[0]))
242 print("phi = {:.2f} graus".format(ret.x[0]))
243 #print("nu  = {:.2f}".format(ret.x[2]))
244 print("t   = {:.2f} MPa".format(ret.x[1]))
245 print("Em  = {:.2f} GPa".format(ret.x[2]))
246 print("R2 min: {:.4f}".format(ret.fun))
247
248 # Ajuste para mostrar o tempo decorrido corretamente
249 if tf - t0 >= 24 * 3600:
250     dia  = int((tf - t0) / 24 / 3600)
251     from operator import mod
252     segundos = mod((tf - t0), 24 * 3600)
253     print("Tempo Total de Processamento: {} d - {}".format(dia,
254           time.strftime("%H:%M:%S", time.gmtime(tf - t0))))
255 else:
256     print("Tempo Total de Processamento: {}".format(time.
    strftime("%H:%M:%S", time.gmtime(tf - t0))))
```

PUC-Rio - Certificação Digital Nº 1712747/CA