



Universidade de Brasília  
Instituto de Geociências - IG

**Desenvolvimento de metodologia de tomografia sísmica  
para aplicações rasas**

Danilo Portela de Oliveira

Orientador: Marcelo Peres Rocha

Coorientador: Giuliano Sant'Anna Marotta

Brasília, XX de Maio de 2022

Danilo Portela de Oliveira

**Desenvolvimento de metodologia de tomografia sísmica  
para aplicações rasas**

Monografia submetida ao curso de graduação em Geofísica da Universidade de Brasília como requisito para obtenção do Título de Bacharel em Geofísica.

Orientador: Marcelo Peres Rocha  
Coorientador: Giuliano Sant'Anna Marotta

Brasília, XX de Maio de 2022

## Agradecimentos

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

*“If I have seen further it is by standing on the shoulders of Giants.”*  
*Sir Isaac Newton*

## Resumo

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

### Palavras-chave:

...

## Abstract

Lore ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lore ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lore ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lore ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lore ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lore ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

## Keywords:

...

## Lista de Figuras

1	O tempo de percurso das ondas acústicas (linhas tracejadas) através das linhas e colunas de uma matriz quadrada de tijolos é medido com a fonte acústica <b>S</b> e o receptor <b>R</b> colocados nas bordas do quadrado. O problema inverso é inferir as propriedades acústicas dos tijolos (que são assumidas como homogêneas). Menke (1984).	16
2	(a) Problema Direto versus Problema Inverso. (b) O problema inverso visto com duas etapas. Gabriela Félix Brião (2005).	20
3	Gráfico de materiais de acordo com a velocidade. Assim, com a solução obtida pela tomografia de tempo de percurso, os geólogos e geofísicos podem inferir que tipo de material existe no subsolo. Gabriela Félix Brião (2005).	20
4	Modelos. Gabriela Félix Brião (2005).	21
5	Tempo de percurso para a $i$ -ésima frente de onda, onde está sendo utilizado o modelo discretizado com vagarosidade constante em cada pixel. Gabriela Félix Brião (2005).	22
6	Diagrama de um experimento tomográfico. A matriz $M$ geralmente é esparsa, pois existem células por onde não passam nenhum raio do experimento. Gabriela Félix Brião (2005).	23
7	Traçado de raios. Discretização do espaço em 49 pixels. <i>Cross-hole array</i> . Imhof e Calvo (2018).	24
8	Matriz de percurso dos raios e significado de cobertura espacial. Imhof e Calvo (2018).	25
9	Gráficos de cobertura espacial para pixels. (a) $10 \times 10$ pixels, (b) $20 \times 20$ pixels. Imhof e Calvo (2018).	26
10	Gráficos de cobertura espacial Ipixel. (a) $10 \times 10$ ipixels, (b) $20 \times 20$ ipixels (calculados a partir de uma matriz de cobertura espacial uniforme de base de $100 \times 100$ pixels). Imhof e Calvo (2018).	26
11	Área de estudo nos fundos do Observatório Sismológico no Campus Darcy Ribeiro da Universidade de Brasília (UnB). A) Delimitando a área de estudo para a implantação do alvo. B) Imagem de satélite do Google Earth mostrando a localização da área de estudo (em vermelho).	29
12	Mapa geológico do Distrito Federal. Modificado de Neumann (2012).	30
13	Mapa de solos da Bacia do Lago Paranoá. Modificado de Menezes (2010).	32
14	Arquitetura de software do pyGIMLI ilustrando seus componentes e diferentes níveis de abstração. A biblioteca Python foi construída sobre um núcleo C++ e várias dependências externas (caixa amarela). (Para interpretação das referências à cor nesta legenda da figura, o leitor deve consultar a versão web <a href="https://www.pygimli.org/design.html#sec-design">https://www.pygimli.org/design.html#sec-design</a> ).	35
15	Esquema de inversão generalizada. Operadores de encaminhamento já implementados ou personalizados podem ser usados para fornecer a função de resposta específica do problema e sua Jacobiana. Várias estratégias estão disponíveis para regularizar o problema inverso.	36
16	Fluxo de operação geral com as etapas usadas em cada módulo: Geração de modelos sintéticos de tomografia sísmica (InTTraPy Sintético) e processamento de dados de tempo de percurso gerando modelos reais de tomografia sísmica (InTTraPy Campo).	43

## **Lista de Tabelas**

1	Parâmetros que devem ser informados para gerar os modelos sintéticos. Observação: posteriormente, será feito uma atualização para que o usuário possa modelar outras geometrias para alvo. . . . .	42
2	Parâmetros que devem ser informados para gerar os modelos reais. . . . .	44

## **Lista de Símbolos**

# Sumário

<b>Lista de Figuras</b>	<b>7</b>
<b>Lista de Tabelas</b>	<b>8</b>
<b>Sumário</b>	<b>10</b>
<b>1 Introdução</b>	<b>12</b>
<b>2 Fundamentação Teórica</b>	<b>13</b>
2.1 Formulando problemas inversos . . . . .	13
2.1.1 Forma linear implícita . . . . .	14
2.1.2 Forma explícita . . . . .	14
2.1.3 Forma linear explícita . . . . .	14
2.1.4 Exemplos de formulação de problemas inversos . . . . .	14
2.2 O Problema Linear Inverso . . . . .	16
2.3 Tomografia . . . . .	16
2.3.1 Tomografia sísmica por tempo de percurso . . . . .	17
2.3.2 Reconstrução na tomografia sísmica de tempo de percurso . . . . .	18
2.3.3 Problema Direto . . . . .	19
2.3.4 Problema Inverso . . . . .	19
2.4 Modelos (Representação da Estrutura) . . . . .	21
2.5 Matriz de percurso dos raios . . . . .	23
2.6 Matriz de cobertura espacial . . . . .	23
2.7 Condicionamento de uma matriz . . . . .	27
2.8 Regularização . . . . .	27
<b>3 Materiais e Métodos</b>	<b>29</b>
3.1 Área de Estudo . . . . .	29
3.1.1 Geologia . . . . .	30
3.1.2 Pedologia . . . . .	31
3.2 Frente teórica-computacional . . . . .	33
3.2.1 Recursos computacionais . . . . .	33
3.2.2 pyGIMLi: Uma biblioteca de código aberto para modelagem e inversão em Geofísica . . . . .	34
3.2.3 Estrutura do <i>Software</i> . . . . .	40
3.3 Etapa experimental - Aquisição em local com alvo conhecido . . . . .	44
<b>4 Resultados e Discussão</b>	<b>46</b>
4.1 Dados sintéticos - Estudos de casos . . . . .	46
4.2 Dados de campo - Processamento e inversão dos dados sísmicos . . . . .	47
<b>5 Conclusões</b>	<b>48</b>
<b>Referências Bibliográficas</b>	<b>49</b>



# 1 Introdução

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

## 2 Fundamentação Teórica

### 2.1 Formulando problemas inversos

O ponto de partida na maioria dos problemas inversos é uma descrição dos dados. Como na maioria dos problemas inversos os dados são simplesmente uma tabela de valores numéricos, um vetor fornece um meio conveniente de sua representação (Menke, 1984).

Se as medições  $N$  forem realizadas em um experimento específico, por exemplo, pode-se considerar esses números como os elementos de um vetor  $\mathbf{d}$  de comprimento  $N$ . Da mesma forma, os parâmetros do modelo podem ser representados como os elementos de um vetor  $\mathbf{m}$ , que é de comprimento  $M$ .

$$\begin{aligned} \text{dados : } \mathbf{d} &= [d_1, d_2, d_3, d_4, \dots, d_N]^T \\ \text{parâmetros do modelo : } \mathbf{m} &= [m_1, m_2, m_3, m_4, \dots, m_N]^T \end{aligned} \quad (1)$$

Aqui  $T$  significa a matriz transposta.

A afirmação básica de um problema inverso é que os parâmetros do modelo e os dados estão de alguma forma relacionados. Essa relação é chamada de *modelo*. Normalmente, o modelo assume a forma de uma ou mais fórmulas que os dados e os parâmetros do modelo devem seguir.

Se, por exemplo, alguém estivesse tentando determinar a densidade de um objeto medindo sua massa e volume, haveria dois dados - massa e volume (digamos,  $d_1$  e  $d_2$ , respectivamente) - e um parâmetro de modelo desconhecido, densidade (digamos,  $m_1$ ). O modelo seria a afirmação de que densidade vezes volume é igual a massa, o que pode ser escrito de forma compacta pela equação vetorial  $d_2m_1 = d_1$ .

Em situações mais realistas, os dados e os parâmetros do modelo são relacionados de maneiras mais complicadas. Mais geralmente, os dados e os parâmetros do modelo podem estar relacionados por uma ou mais equações implícitas, como

$$\begin{aligned} f_1(\mathbf{d}, \mathbf{m}) &= 0 \\ f_2(\mathbf{d}, \mathbf{m}) &= 0 \\ &\vdots \\ f_L(\mathbf{d}, \mathbf{m}) &= 0 \end{aligned} \quad (2)$$

Onde  $L$  é o número de equações. As Equações em (2) da página 13 relativos à medição de densidade,  $L = 1$  e  $d_2m_1 = d_1$  constituiria a única equação da forma  $f_1(\mathbf{d}, \mathbf{m}) = 0$ . Essas equações implícitas, que podem ser escritas de forma compacta como a equação vetorial  $\mathbf{f}(\mathbf{d}, \mathbf{m}) = 0$ , resume o que se sabe sobre como os dados medidos e os parâmetros desconhecidos do modelo estão relacionados. O propósito da teoria inversa é resolver, ou “inverter”, essas equações para os parâmetros do modelo, ou quaisquer tipos de respostas que possam ser possíveis ou desejáveis em qualquer situação.

Nenhuma afirmação é feita de que as equações  $\mathbf{f}(\mathbf{d}, \mathbf{m}) = 0$  contenham informações suficientes para especificar os parâmetros do modelo de forma única ou que sejam consistentes. Um dos propósitos da teoria inversa é responder a esses tipos de perguntas e fornecer meios de lidar com os problemas que elas implicam. Em geral,  $\mathbf{f}(\mathbf{d}, \mathbf{m}) = 0$  pode consistir em funções arbitrariamente complicadas (não lineares) dos dados e parâmetros do modelo. Em muitos problemas, no entanto, a equação assume uma das várias formas simples.

### 2.1.1 Forma linear implícita

A função  $\mathbf{f}$  é linear em ambos os parâmetros de dados e modelo e, portanto, pode ser escrita como a equação matricial (Equações em (2) da página 13). O Problema Linear Inverso

$$\mathbf{f}(\mathbf{d}, \mathbf{m}) = 0 = \mathbf{F} \begin{bmatrix} \mathbf{d} \\ \mathbf{m} \end{bmatrix} \quad (3)$$

onde  $\mathbf{F}$  é uma matriz  $L \times (M + N)$ .

### 2.1.2 Forma explícita

Em muitos casos, é possível separar os dados dos parâmetros do modelo e, assim, formar  $L = N$  equações que são lineares nos dados (mas ainda não lineares nos parâmetros do modelo por meio de uma função vetorial  $\mathbf{g}$ ).

$$\mathbf{f}(\mathbf{d}, \mathbf{m}) = 0 = \mathbf{d} - \mathbf{g}(\mathbf{m}) \quad (4)$$

### 2.1.3 Forma linear explícita

Na forma linear explícita, a função  $\mathbf{g}$  também é linear, levando à equação matricial  $N \times M$  (onde  $L = N$ )

$$\mathbf{f}(\mathbf{d}, \mathbf{m}) = 0 = \mathbf{d} - \mathbf{G}\mathbf{m} \quad (5)$$

Usar esta forma é equivalente a dizer que a matriz  $\mathbf{F}$  na Seção 2.1.1 é uma matriz diagonal.

$$\mathbf{F} = \begin{bmatrix} -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} \end{bmatrix} \quad (6)$$

### 2.1.4 Exemplos de formulação de problemas inversos

#### **Exemplo 1 | Ajustando uma linha reta**

Suponha que  $N$  medições de temperatura  $T_i$  sejam feitas em profundidades  $z_i$  na terra (Menke, 1984). Os dados então são, um vetor  $\mathbf{d}$  de  $N$  medições de temperatura, onde  $\mathbf{d} = [T_1, T_2, T_3, \dots, T_N]^T$ . As profundidades  $z_i$ , não são, estritamente falando, dados. Em vez disso, eles fornecem algumas informações auxiliares que descrevem a geometria do experimento. Essa distinção será melhor esclarecida a seguir.

Suponha que assumimos um modelo no qual a temperatura é uma função linear da profundidade:  $T = a + bz$ . O coeficiente linear  $a$  e o coeficiente angular  $b$  formam então os dois parâmetros do modelo do problema,  $\mathbf{m} = [a, b]^T$ . De acordo com o modelo, cada observação de temperatura deve satisfazer  $T = a + bz$ :

$$\begin{aligned} T_1 &= a + bz_1 \\ T_2 &= a + bz_2 \\ &\vdots \\ T_N &= a + bz_N \end{aligned} \quad (7)$$

Essas equações (7), página 14, podem ser organizadas como a equação matricial  $\mathbf{Gm} = \mathbf{d}$ :

$$\begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_N \end{bmatrix} = \begin{bmatrix} 1 & z_1 \\ 1 & z_2 \\ \vdots & \vdots \\ 1 & z_N \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \quad (8)$$

### **Exemplo 2 | Tomografia acústica**

Suponha que uma parede seja montada a partir de um arranjo retangular de tijolos (Figura 1, página 16) e que cada tijolo seja composto de um tipo diferente de argila (Menke, 1984). Se as velocidades acústicas das diferentes argilas diferem, pode-se tentar distinguir os diferentes tipos de tijolos medindo o tempo de viagem do som através das várias fileiras e colunas de tijolos na parede. Os dados deste problema são  $N = 8$  medições de tempo de percurso,  $\mathbf{d} = [T_1, T_2, T_3, \dots, T_8]^T$ .

O modelo assume que cada tijolo é composto de um material uniforme e que o tempo de percurso do som através de cada tijolo é proporcional à largura e altura do tijolo. O fator de proporcionalidade é a vagarosidade (*slowness*) do tijolo  $s_i$ , dando assim  $M = 16$  parâmetros do modelo  $\mathbf{m} = [s_1, s_2, s_3, \dots, s_{16}]^T$ , onde a ordenação está de acordo com o esquema de numeração da figura como

$$\begin{aligned} \text{Linha1 : } T_1 &= hs_1 + hs_2 + hs_3 + hs_4 \\ \text{Linha2 : } T_2 &= hs_5 + hs_6 + hs_7 + hs_8 \\ &\vdots \\ \text{Coluna4 : } T_8 &= hs_4 + hs_8 + hs_{12} + hs_{16} \end{aligned} \quad (9)$$

e a equação matricial é

$$\begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_8 \end{bmatrix} = h \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_{16} \end{bmatrix} \quad (10)$$

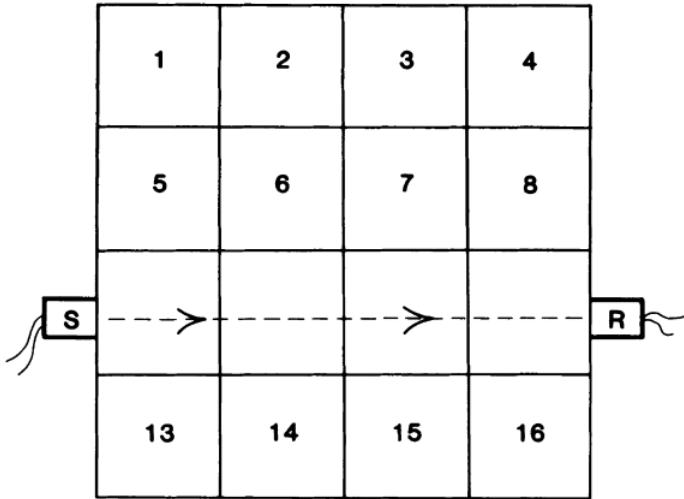


Figura 1: O tempo de percurso das ondas acústicas (linhas tracejadas) através das linhas e colunas de uma matriz quadrada de tijolos é medido com a fonte acústica **S** e o receptor **R** colocados nas bordas do quadrado. O problema inverso é inferir as propriedades acústicas dos tijolos (que são assumidas como homogêneas). Menke (1984).

## 2.2 O Problema Linear Inverso

Os problemas inversos mais simples e mais bem compreendidos são aqueles que podem ser representados com a equação linear explícita  $\mathbf{Gm} = \mathbf{d}$ . Esta equação, portanto, forma a base do estudo da teoria inversa discreta. Muitos problemas inversos importantes que surgem nas ciências físicas envolvem precisamente essa equação. Outros, embora envolvam equações mais complicadas, muitas vezes podem ser resolvidos por meio de aproximações lineares.

A matriz  $\mathbf{G}$  é chamada de *data kernel*, em analogia à teoria das equações integrais, na qual os análogos aos vetores de parâmetros de dados e modelos são duas funções contínuas  $d(x)$  e  $m(x)$ , onde  $x$  é alguma variável independente. As duas funções estão relacionadas pela equação

$$d(x) = \int G(x, \xi)m(\xi)d\xi \quad (11)$$

onde a função  $G(x, \xi)$  é o *kernel*, ou função de Green, da equação integral. A solução de problemas desse tipo está dentro do escopo da teoria inversa contínua.

## 2.3 Tomografia

O objetivo basal da Geofísica é obter informações a cerca dos parâmetros físicos das rochas que constituem o interior da Terra a partir de dados medidos em superfície, em poços ou em levantamentos aéreos (Reis Rodrigues e Bassrei, 2015). Na modelagem direta, aplicam-se as equações que descrevem as leis da física sobre um modelo previamente definido para a obtenção de dados de resposta referentes aos parâmetros físicos desse tal modelo. Em contra partida, no problema inverso, já se possui os dados de resposta, aqueles obtidos através de levantamentos, e, portanto, o objetivo agora é a estimativa de

um modelo que, quando aplicado a uma determinada relação físico-matemática, melhor se adéque aos dados medidos.

A tomografia sísmica é uma metodologia de inferência de parâmetros numéricos, usada na Geofísica, que extrai informações contidas em registros sísmicos para estimar modelos bidimensionais ou tridimensionais do interior da Terra (Rawlinson et al., 2010). Em geral, requer a solução de um problema inverso para se obter um modelo de velocidades sísmicas que seja consistente com as observações de campo. Desde que seja possível estabelecer um modelo aproximado  $\mathbf{d} = g(\mathbf{m})$ , entre o vetor de dados sísmicos  $\mathbf{d}$  e o vetor de parâmetros do modelo sísmico  $\mathbf{m}$  - de modo que, para um dado modelo  $\mathbf{m}$ , seja possível prever  $\mathbf{d}$  - procura-se, na tomografia sísmica, encontrar  $\mathbf{m}$  tal que  $\mathbf{d}_{obs} = g(\mathbf{m})$ , onde  $\mathbf{d}_{obs}$  é o vetor de dados observados.

As relações do tipo  $\mathbf{d} = g(\mathbf{m})$ , em sua grande maioria, são não-lineares e não solucionáveis analiticamente (Reis Rodrigues e Bassrei, 2015), por isso, busca-se a linearização dessas relações através de métodos numéricos. Com isso, devido a essa não-linearidade do problema inverso, a superfície da função-objetivo dos tempos de percurso pode não vir a ser simples, bem comportada ou com um único mínimo bem definido.

Devido, também, ao caráter discreto dos dados geofísicos, faz-se necessário o uso de uma formulação matricial para o tratamento desses dados (Reis Rodrigues e Bassrei, 2015). O objetivo agora, então, é criar uma aproximação linearizada  $\mathbf{Gm} = \mathbf{d}$ , onde  $\mathbf{d}$  é um vetor  $p$ -dimensional que carrega os dados medidos nos levantamentos,  $\mathbf{m}$  é um vetor  $n$ -dimensional que carrega os parâmetros do modelo o qual se quer estimar a partir das medidas, e  $\mathbf{G}$  é uma matriz de dimensões  $p$  por  $n$  que, se inversível, nos levaria facilmente a solução do problema através da relação  $\mathbf{m} = \mathbf{G}^{-1}\mathbf{d}$ . Todavia, na quase totalidade dos problemas geofísicos, os problemas são mal postos (Reis Rodrigues e Bassrei, 2015), ou seja, não obedecem aos critérios de existência, unicidade e estabilidade simultaneamente; e  $\mathbf{Gm} = \mathbf{d}$  constitui-se num sistema sobre-determinado ( $m > n$ ), onde  $\mathbf{G}$  não é, à rigor, inversível; e, portanto, se faz necessário o uso de métodos numéricos mais complexos e, também, a consulta de informações geológicas à priori que, de alguma forma, limite os graus de liberdade do problema.

A tomografia sísmica possui grande importância no processamento sísmico. Através dela, é possível extrair diversas informações dos registros sísmicos, os quais se incluem tempos de percurso, amplitudes, conteúdo de frequências ou, até mesmo, a forma total da onda. Ela pode ser considerada, por exemplo, um complemento natural para a migração sísmica, pois oferece um meio de estimar a velocidade e a profundidade de interfaces usando tempos de percurso e, com menos frequência, amplitudes de espalhamento geométrico e coeficientes de reflexão/transmissão (Reis Rodrigues e Bassrei, 2015).

### 2.3.1 Tomografia sísmica por tempo de percurso

A inversão sísmica de tempo de percurso para a estrutura de velocidades é um problema não linear, uma vez que os raios sísmicos, atuando como caminhos integrais em uma inversão tomográfica, dependem também da velocidade média (Worthington, 1984; Zhou e Greenhalgh, 2003). Em vez de determinar os caminhos de raios desconhecidos e a estrutura de velocidade desconhecida simultaneamente, muitas vezes usa-se uma solução linearizada de forma iterativa para resolver as perturbações do modelo, em vez dos parâmetros do modelo diretamente (Bording et al., 1987). As perturbações do modelo são consideradas pequenas o suficiente para que se possa considerar a relação entre as perturbações do modelo e os resíduos correspondentes como linear. Uma vez que as per-

turbações do modelo tenham sido encontradas, elas são adicionadas ao modelo atual para produzir o modelo para a iteração seguinte até que o resultado converja.

Na prática, os dados sísmicos às vezes contêm uma quantidade considerável de ruído. Para dados de tempo de percurso, a precisão de leitura finita é a principal fonte de erros de dados. Para reduzir os erros, um método de pré-processamento é calcular a média dos dados de tempo de percurso de caminhos de raios semelhantes Röhm et al. (2000), mas esse método pode funcionar apenas em tomografia telessísmica, pois os erros são relativamente pequenos em comparação com os dados reais do tempo de percurso. Para mitigar os erros de dados, Wang et al. (2000) separaram os dados de tempo de percurso com ruído usando uma regressão ponderada localmente para se livrar dos valores discrepantes ou reduzir o peso dos grandes erros de separação. Para reduzir o efeito de erros de dados durante a inversão, Scales et al. (1988) usaram coeficientes de ponderação em função do tempo de percurso residual em um esquema de mínimos quadrados iterativamente reponderado. Esses pesos baseados em resíduos, no entanto, são subjetivos, uma vez que os erros de dados na prática podem ter uma distribuição não Gaussiana e, portanto, o resíduo de dados pode ser enviesado. Cao e Greenhalgh (1995) montaram um esquema de inversão não linear baseado em erro relativo para dados sísmicos de *crosshole* que tenta superar o viés dos métodos de mínimos quadrados que tendem a enfatizar raios com maiores tempos de percurso.

### 2.3.2 Reconstrução na tomografia sísmica de tempo de percurso

O problema de reconstruir imagens por projeções, isto é, reconstruir uma função através de suas integrais ao longo de retas surgiu independentemente em diversos ramos da ciência tais como na Geofísica, Astrofísica, Medicina e dentre outros.

Provavelmente os exemplos que causaram maior impacto na vida moderna foram em prospecção sísmica e na tomografia computadorizada voltada para diagnósticos clínicos.

O termo tomografia surge do prefixo grego “tomo” que quer dizer fatia, o que nos sugere uma reconstrução em 2-D. Mas a palavra já é utilizada rotineiramente para se referir a reconstrução de imagens em 3-D, sobretudo pelos sismólogos e radiólogos.

A tomografia sísmica tem como preocupação, a reconstrução de imagens de estruturas em subsuperfície. Dentre as técnicas existentes focalizaremos a que utiliza dados de tempo de percurso (Gabriela Félix Brião, 2005).

Iniciaremos com a descrição do Princípio de Fermat. Neste contexto, é natural introduzir a vagarosidade, ou seja, a inversa da velocidade.

Dada  $s$  uma distribuição contínua da vagarosidade  $s(x)$ , o *tempo de percurso* de um sinal ao longo de um possível caminho que liga a fonte que o emitiu ao receptor é dado pela Equação (12) da página 18:

$$\tau^P(s) = \int_P s(x) dl^P = \int_P \frac{1}{v(x)} dl^P \quad (12)$$

Onde  $dl^P$  denota o comprimento de arco ao longo do caminho  $P$ . Denotemos por  $\tau$  o conjunto de todos os possíveis caminhos ligando a fonte ao receptor.

O Princípio de Fermat diz que o caminho físico percorrido por uma onda entre dois pontos é aquele que minimiza o tempo de percurso.

$$\tau^*(s) = \min_{P \in \{\tau\}} \tau^P(s) \quad (13)$$

O funcional  $P \mapsto \tau^p(s)$  de tempo de percurso é estacionário com respeito a pequenas pertubações no caminho de Fermat  $P^*(s)$  no sentido do cálculo das variações (Axelsson, 1994). Observe que a Equação (13) da página 18 depende de forma não linear em  $s$ , como consequência do processo de minimização.

Uma forma de obter dados na tomografia sísmica é feita aproveitando a existência de poços já perfurados. Isto é, colocando transmissores em um dos poços e receptores em outro de maneira que são emitidas ondas entre os poços. Tais ondas podem ser sísmicas ou eletromagnéticas. Através da utilização de receptores apropriados mede-se o tempo de chegada das mesmas.

Esses dados, apesar de imprecisos e ruidosos, podem ser utilizados para obtermos informações sobre a composição do subsolo e a presença de hidro-carbonetos.

### 2.3.3 Problema Direto

Dentro de uma aproximação aceitável para muitas finalidades em Geofísica, podemos modelar as ondas sísmicas como soluções da equação diferencial parcial:

$$\partial_t^2 \phi - c^2(x) \Delta \phi = g(x, t) \quad (14)$$

onde  $g(x, t)$  é a intensidade da perturbação num determinado ponto  $x$  e tempo  $t$ . Temos que  $\phi(x, t)$  é a intensidade da onda no tempo  $t$  e posição  $x$ . Esta equação deve ser complementada com condições de contorno apropriadas.

O problema direto consiste em resolver a equação dado  $g$ , isto é, encontrar a solução  $\phi$ . No caso da tomografia por tempo de percurso, a preocupação é determinar o tempo de percurso da frente de onda e o caminho percorrido pela mesma entre fonte e receptor. Para saber mais sobre as técnicas de resolução desse tipo de problema direto ver (Rawlinson e Sambridge, 2003).

### 2.3.4 Problema Inverso

Para o problema inverso (Figura 2a, página 20), queremos obter informações sobre os coeficientes da equação utilizando dados sobre suas soluções em regiões distintas. As técnicas de problemas inversos são de grande interesse na prospecção sísmica, pois têm por objetivo, determinar o interior da região em estudo somente com base em informação parcial dos dados no exterior da mesma. Assim, nos propomos a reconstruir os valores de  $c(x)$  com base na solução da equação da onda medida na fronteira que delimita a região de interesse. Com essas informações em mãos, os geólogos e geofísicos podem inferir que tipo de material existe no subsolo estudado fazendo uso de informações como por exemplo na Figura 3 (página 20).

Na inversão linear na tomografia por tempo de percurso, assumimos à priori que sabemos o traçado dos feixes que ligam fonte a receptor, o que é justificado por uma aproximação linear que ignora a dependência que os caminhos possuem da distribuição da vagarosidade (Princípio de Fermat).

Na Figura 2a (página 20) descrevemos simplificadamente uma comparação entre o problema direto e o problema inverso. O problema inverso é relativamente mais complicado, uma vez que, em problemas reais, fixados os dados, podemos construir infinitos modelos que se adéquam a estes mesmos dados. No problema inverso, muitas vezes não há essa unicidade levando dos dados ao modelo. Assim, chegamos a um esquema mais adequado à realidade na Figura 2b (página 20).

A não unicidade do problema inverso pode ser explicada pelo fato de possuirmos somente uma quantidade finita de dados coletados para obter um modelo que muitas vezes é uma função contínua de suas variáveis, o que significa, que o mesmo possui infinitos graus de liberdade. Por causa dessa limitação física da finitude dos dados, o modelo que alcançamos através dos dados coletados não é necessariamente o que modela a realidade.

São necessários dois passos na inversão para chegarmos a um modelo mais próximo da realidade. Isto é representado na Figura 2b (página 20). Vale salientar que em última análise o modelo verdadeiro não é sabido em problemas reais.

O primeiro passo seria então reconstruir um modelo  $m'$  utilizando os dados  $d$ . Uma vez feito isto, determinamos que propriedades o modelo  $m'$  preserva do modelo real  $m$  e que tipo de erros e ruídos estão associados a ele, ou seja, fazemos uma avaliação do modelo.

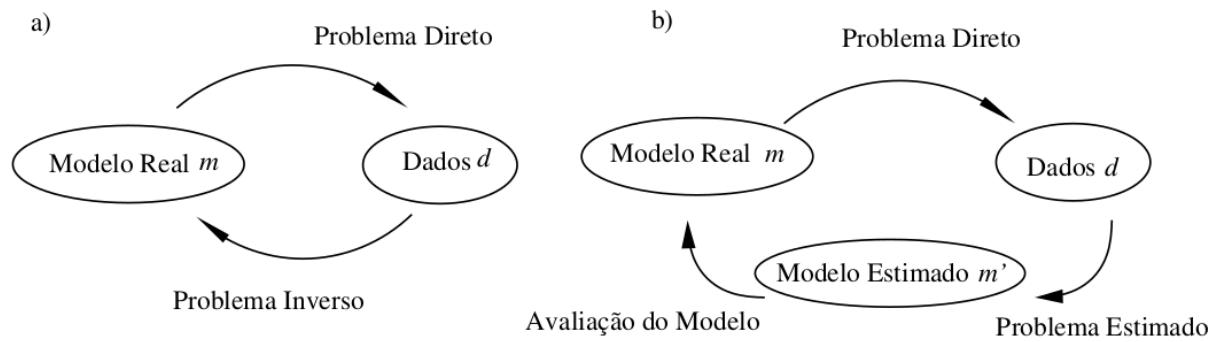


Figura 2: (a) Problema Direto versus Problema Inverso. (b) O problema inverso visto com duas etapas. Gabriela Félix Brião (2005).

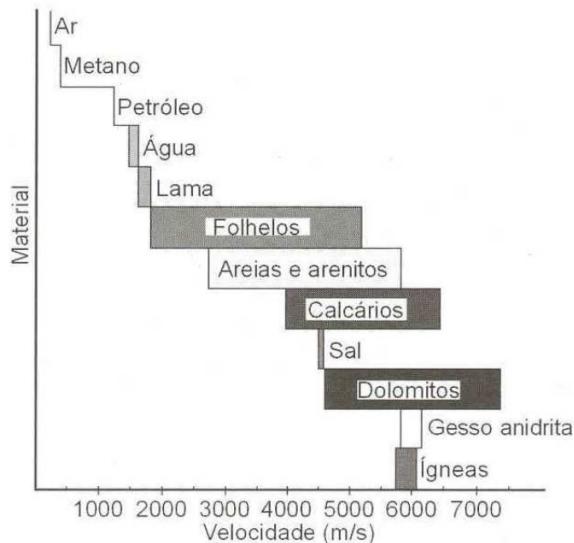


Figura 3: Gráfico de materiais de acordo com a velocidade. Assim, com a solução obtida pela tomografia de tempo de percurso, os geólogos e geofísicos podem inferir que tipo de material existe no subsolo. Gabriela Félix Brião (2005).

## 2.4 Modelos (Representação da Estrutura)

Mostraremos duas maneiras de parametrizar a vagarosidade (Gabriela Félix Brião, 2005). O mais simples seria dividir a região em pequenos blocos (denominados pixels no caso 2-D e voxels no caso 3-D) e atribuir valores constantes à vagarosidade em cada bloco. Isto pode ser visto na Figura 4a (página 21).

Uma alternativa a este modelo é definir vagarosidade nos vértices da malha formada pela divisão da região em blocos (Figura 4b, página 21). Essa definição seria formulada em conjunto com uma função de interpolação. Um exemplo ilustrativo disto seria no contexto de tomografia local de terremotos, tal que para cada vértice  $(x, y, z)$  é utilizada uma interpolação trilinear (figura 4c, página 21):

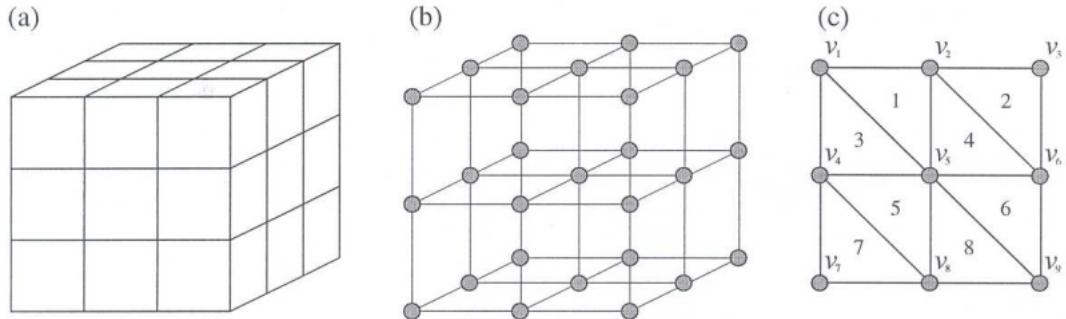


Figura 4: Modelos. Gabriela Félix Brião (2005).

$$v(x, y, z) = \sum_{i=1}^2 \sum_{j=1}^2 \sum_{k=1}^2 V(x_i, y_j, z_k) \left(1 - \left|\frac{x - x_i}{x_2 - x_1}\right|\right) \left(1 - \left|\frac{y - y_j}{y_2 - y_1}\right|\right) \left(1 - \left|\frac{z - z_k}{z_2 - z_1}\right|\right) \quad (15)$$

onde  $V(x_i, y_j, z_k)$  são os valores da velocidade nos oito vértices que cercam o vértice  $(x, y, z)$ .

Para este trabalho estamos interessados em entender o primeiro modelo acima (Figura 4a, página 21). Sendo assim, considere  $t_1, \dots, t_m$  conjunto de tempos de percurso entre fonte e receptor. Dado um modelo com  $n$  células, podemos escrever,

$$t_i = \sum_{j=1}^n l_{ij} s_j, \quad (16)$$

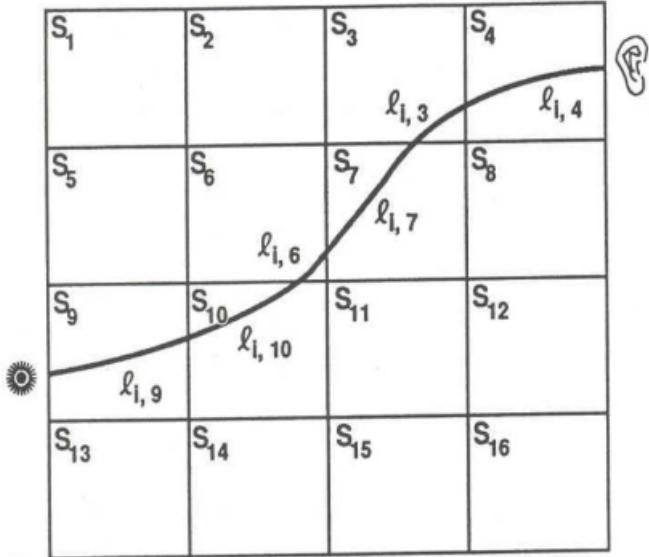


Figura 5: Tempo de percurso para a  $i$ -ésima frente de onda, onde está sendo utilizado o modelo discretizado com vagarosidade constante em cada pixel. Gabriela Félix Brião (2005).

Ou melhor,  $Ms = t$ . Onde  $M$  é a matriz formada pelo comprimento  $l_{ij}$  do  $i$ -ésimo raio que passa pela  $j$ -ésima célula e  $s$  é a vagarosidade (a nossa incógnita). Observe que

$$l_{ij} = \frac{\partial t_i}{\partial s_j} \quad (17)$$

e assim,

$$t_i = \frac{\partial t_i}{\partial s_1} s_1 + \frac{\partial t_i}{\partial s_2} s_2 + \cdots + \frac{\partial t_i}{\partial s_n} s_n. \quad (18)$$

Assim, discretizando o domínio da vagarosidade obtemos um sistema de equações lineares, onde a matriz do sistema é muito esparsa (A densidade de uma matriz é o número de elementos não nulos dividido pelo total de elementos da matriz. Se esse número for muito pequeno essa matriz é dita esparsa) porque cada raio intersecta somente uma pequena fração dos voxels da discretização (ver Figura 5, página 22). Então para os pixels em 2-D, cada raio intersecta algo da ordem de  $m * n$  pixels em uma malha  $m \times n$ . Isso torna o problema particularmente atrativo para a utilização de soluções iterativas.

A matriz  $M$  contém todas as informações físicas e matemáticas que escolhemos para o modelo no problema dado. Assim, no caso da tomografia por tempo de percurso, a matriz  $M$  terá como suas componentes os dados do comprimento das trajetórias.

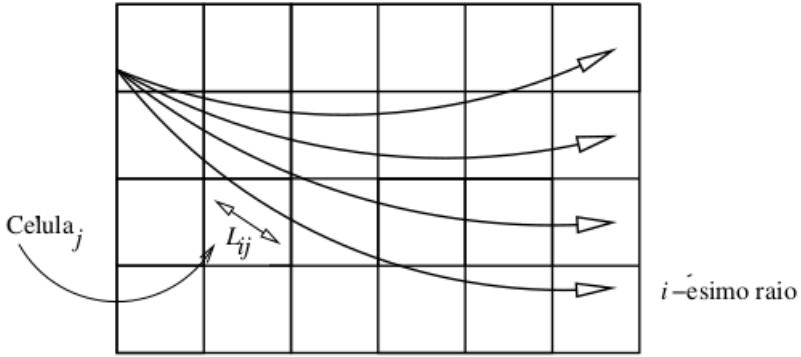


Figura 6: Diagrama de um experimento tomográfico. A matriz  $M$  geralmente é esparsa, pois existem células por onde não passam nenhum raio do experimento. Gabriela Félix Brião (2005).

## 2.5 Matriz de percurso dos raios

As entradas em  $\mathbf{S}$  (Figura 8, página 25) são calculadas identificando a interseção de forma individual do raio ‘ $m$ ’ com os limites do pixel ‘ $n$ ’ e computando o comprimento (relação de Pitágoras) ‘ $d$ ’ dentro do elemento (Imhof e Calvo, 2018). Portanto, o tamanho de  $\mathbf{S}$  crescerá com o incremento das medidas e/ou densidade de discretização.

Considerando a Figura 7 (página 24) e tendo ainda em mente a Figura 8 (página 25); o número de fontes e receptores é 7 e então o número total de raios é 49 (constante para esta matriz). Se o meio for dividido em  $7 \times 7 = 49$  pixels,  $\mathbf{S}$  será de 49 linhas ( $M$ , número de raios) por 49 colunas ( $N$ , número de pixels, cujas vagarosidades (*slowness*) são desconhecidas). É relevante notar que o incremento da densidade de pixels ( $N$ ), levará a considerar mais pixels no cálculo dos caminhos de viagem para cada receptor.

Este sistema de equações é *aparentemente* determinado (número de equações igual ao número de incógnitas). A palavra *aparentemente* significa que em alguns casos (especialmente em *cross-hole*, (ver Imhof, 2007) o sistema está mal condicionado e a classificação de  $\mathbf{S} < N, M$ . Isso dará um sistema subdeterminado levando a infinitas soluções. O mesmo ocorrerá, por exemplo, dividindo o meio em mais pixels para melhorar a resolução das imagens. (a limitação de dividir o meio com  $N = M$  é que a resolução é grosseira, pois o tamanho dos pixels é grande).

Incrementar o número de transdutores para melhorar a resolução não é prático e sempre possível; primeiro, devido à quantidade de esforço de levantamento necessário (custo) e, segundo, se os raios estão tão próximos, a condição de matriz  $\mathbf{S}$  aumenta e não necessariamente adiciona informações ao sistema (Santamarina e Fratta, 1998; Fernandez, 2000).

## 2.6 Matriz de cobertura espacial

O tamanho de  $\mathbf{S}$  (matriz de transformação) é  $M$  linhas (número de medições) por  $N$  colunas (número de vagarosidade - *slowness* - de pixel desconhecido). A soma de cada coluna individual de  $\mathbf{S}$  traz o comprimento total percorrido pelos raios em um pixel (Figura 8, página 25). Aplicando esta soma em todas as colunas de  $\mathbf{S}$ , obtém-se um vetor

linha  $1 \times N$  que rearranjado seguindo o padrão geométrico, traz a matriz de cobertura espacial de “ $s$ ” pixels verticais por “ $t$ ” horizontais onde  $N = s + t$ . Considerou-se aqui  $s = t$  (Imhof e Calvo, 2018).

Dois exemplos de matrizes de cobertura espacial de *cross-hole* são representados na Figura 9 (página 26) para 10 pares de fonte-receptor; **(a)** para 100 elementos e **(b)** 400 unidades. As zonas escuras indicam valores mais baixos de cobertura espacial. Isso significa que as informações coletadas para resolver a velocidade ou vagarosidade (*slowness*) do mesmo são menores do que em outras zonas mais claras. Em outras palavras, a precisão para a avaliação dos valores dos pixels não será uniforme. Em virtude de haver mais raios que atravessam um pixel, há mais informações nele (semelhante ao conceito CDP - *Common Depth Point* - em sísmica de reflexão), o setor mais claro no centro da matriz de cobertura espacial representa a resolução e precisão máximas para o pixel/s ali situado. Mas o que acontece quando uma inclusão que está sendo buscada está longe dessa posição? A resolução para localizá-lo será mais pobre (Santamarina e Fratta, 1998).

Uma forma alternativa de dividir o meio para melhorar a resolução nas zonas escuras é proposta: ao invés de separá-lo em pixels de mesmo tamanho e cobertura espacial diferente em cada um; ele será dividido em elementos de igual cobertura espacial e tamanhos individuais distintos e nomeados como ipixels. Devido a este fato, os elementos de **S** serão diferentes. A Figura 10 (página 26) mostra o domínio dividido em duas densidades de ipixels. Os mesmos tons de cor representam a mesma informação em cada elemento.

Qualquer tipo de discretização de elementos para o domínio físico é perfeitamente possível de realizar porque qualquer tipo dela é apenas geométrica e tem por finalidade fazer a matriz **S** e armar o sistema de equações tendo as distâncias percorridas pelos raios.

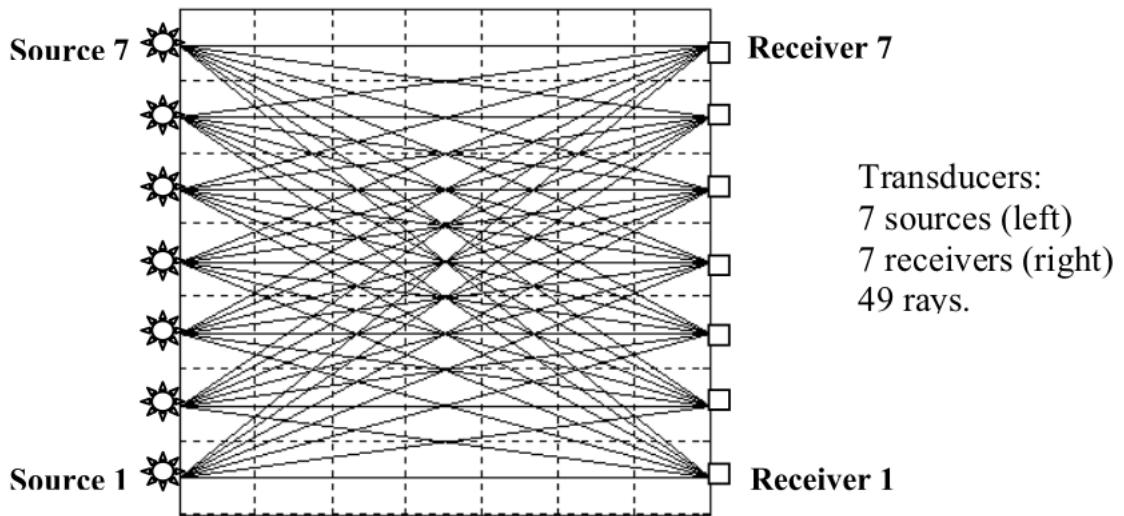
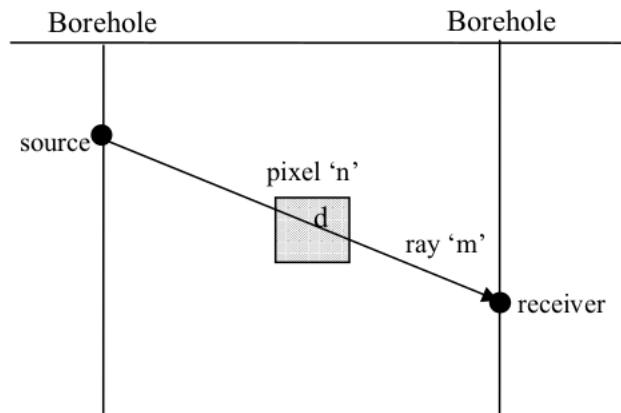


Figura 7: Traçado de raios. Discretização do espaço em 49 pixels. *Cross-hole array*. Imhof e Calvo (2018).



$$\mathbf{S} = \begin{pmatrix} S_{11} & \dots & S_{1n} & \dots & S_{N1} \\ \dots & \dots & \dots & \dots & \dots \\ S_{m1} & \dots & S_{mn} & \dots & S_{mN} \\ \dots & \dots & \dots & \dots & \dots \\ S_{M1} & \dots & S_{Mn} & \dots & S_{MN} \end{pmatrix}$$

$\sum_{m=1}^M S_{mn} = sc_n$

$\mathbf{S}$ : raypath matrix  
 $S_{mn}$ : length travelled by ray 'm' in pixel 'n'  
 $sc_n$ : length travelled by all rays in pixel 'n'.  
 $M$ : number of travel-time measurements  
 $N$ : number of pixels.

Figura 8: Matriz de percurso dos raios e significado de cobertura espacial. Imhof e Calvo (2018).

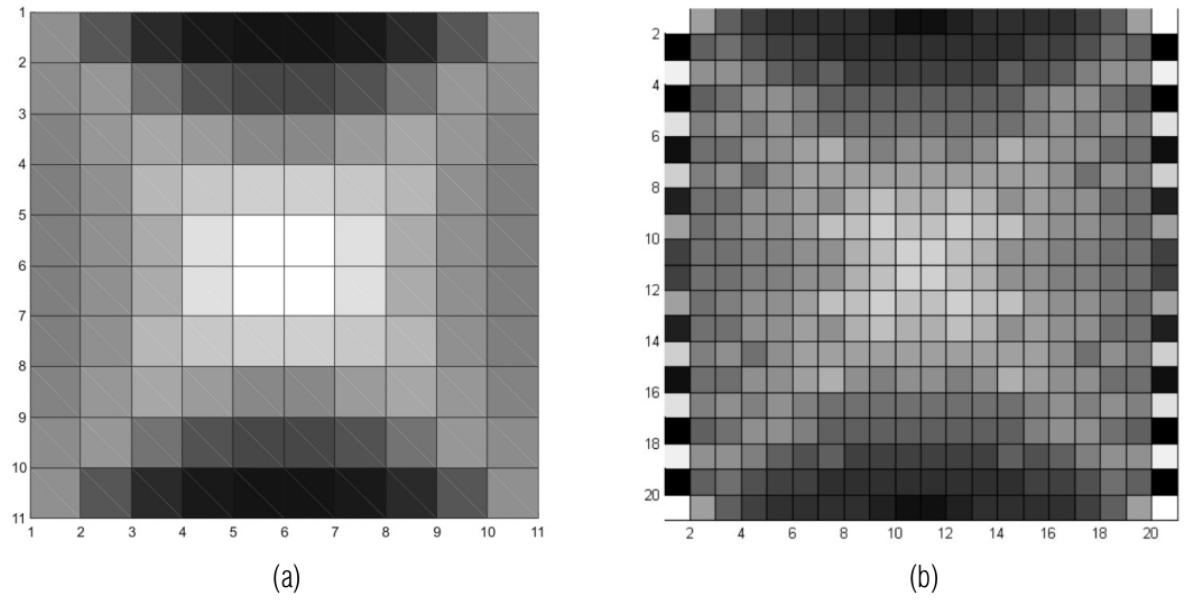


Figura 9: Gráficos de cobertura espacial para pixels. (a)  $10 \times 10$  pixels, (b)  $20 \times 20$  pixels. Imhof e Calvo (2018).

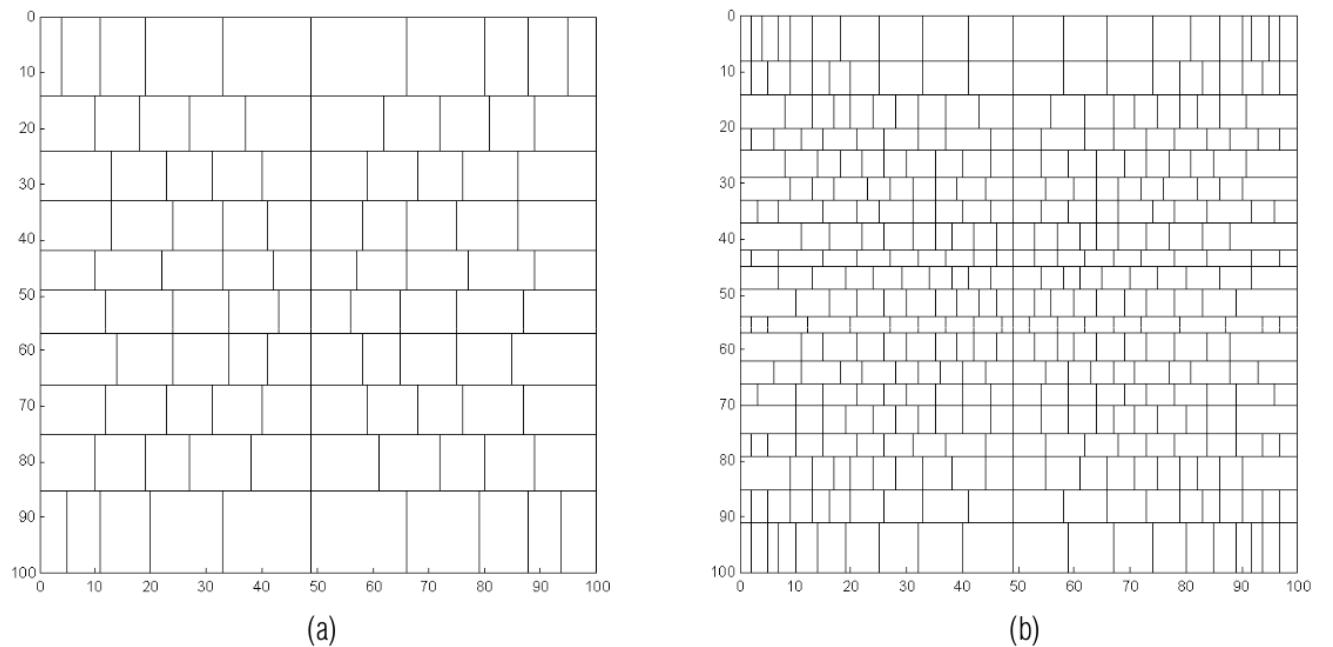


Figura 10: Gráficos de cobertura espacial Ipixel. (a)  $10 \times 10$  ipixels, (b)  $20 \times 20$  ipixels (calculados a partir de uma matriz de cobertura espacial uniforme de base de  $100 \times 100$  pixels). Imhof e Calvo (2018).

## 2.7 Condicionamento de uma matriz

Se houver um alto contraste entre os valores numéricos dos elementos dentro de uma matriz; é possível que a classificação calculada para ela tenha um número errôneo de linhas linearmente independente. Devido a isso, o número de condição  $k$  é uma escolha melhor para estudar o condicionamento de uma matriz (ou seja, quão próximo está de uma matriz de classificação mais baixa) do que sua classificação (Strang, 1980).

$k$  é definido como a razão entre os valores singulares com os valores absolutos máximos e mínimos (Computations, 1989):

$$k = \frac{\max|\lambda_i|}{\min|\lambda_i|} \quad (19)$$

Uma matriz é dita mal condicionada se  $k$  for muito grande. Geometricamente, os valores máximo e mínimo representam os eixos de uma elipse (Branham, 1990). Se a razão tende à unidade, a correlação é nula (independente) e se for muito grande, é quase perfeita (dependente) e portanto mal condicionada. Em outras palavras, mau condicionamento significa que o número de linhas ou colunas linearmente dependentes da matriz tende a aumentar. Isso significa que o número de equações/medidas diminui em relação ao número de incógnitas.

Por fim, os valores singulares dão uma indicação relacionada à confiança e à relação entre as medidas dos fenômenos. Valores pequenos sugerem informações limitadas sobre o parâmetro.

## 2.8 Regularização

Problemas em tomografia geralmente são mal-postos, isto é, problemas que falham, seja na existência de soluções, na unicidade dessas soluções ou mesmo que a solução não depende continuamente dos dados (Gabriela Félix Brião, 2005). Sendo assim, são utilizadas frequentemente técnicas de regularização para dar estabilidade ao problema (Natterer e Wübbeling, 2001). Essas técnicas nos permitem solucionar não o problema original mas sim, um problema similar, porém mais robusto em relação a erros nos dados.

Considere o problema de resolver  $Ms = t$ . Do ponto de vista matemático, no caso de modelos lineares, esses problemas mal-postos se devem geralmente ao fato da matriz  $M$  possuir valores singulares nulos ou muito próximos de zero. Uma das formas de contornar isto seria acrescentar à matriz  $M^T M$  um múltiplo da matriz identidade de tal maneira que essa nova matriz possua somente valores singulares positivos, porém distantes do zero. De fato, considerando  $B = M^T M + \gamma I$ , temos que se  $\gamma \neq 0$ , os autovalores de  $B$  ficam diferentes de zero (positivos).

Feito isto, podemos definir a solução de mínimos quadrados amortecidos do sistema original por:

$$s' = (M^T M + \gamma I)^{-1} M^T t \quad (20)$$

A escolha de um bom parâmetro  $\gamma$  é fundamental nos problemas mal postos. O número  $\gamma$  é chamado de parâmetro de regularização.

A não existência ou a perda de unicidade das soluções se devem ao fato de  $t \notin Im(M)$  ou a não injetividade da transformação  $M$ , respectivamente. Nesses casos, a utilização da pseudo-inversa  $M^\dagger$  é o mais conveniente. A técnica de regularização de Tikhonov

consiste em obter certas transformações dadas denotadas por  $A_\lambda : \mathbb{R}^n \rightarrow \mathbb{R}^m$ ,  $\lambda > 0$ , de tal forma,

$$\lim_{\lambda \rightarrow 0} A_\lambda t = M^\dagger t \quad (21)$$

onde  $M^\dagger$  é a pseudo-inversa da matriz  $M$ .

Seja  $t_\epsilon \in \mathbb{R}^n$  tal que  $\|t - t_\epsilon\| \leq \epsilon$ . E seja  $\lambda(\epsilon)$  tal que, quando  $\epsilon \rightarrow 0$

$$\lambda(\epsilon) \rightarrow 0 \quad (22)$$

e

$$\|A_{\lambda(\epsilon)}\| \epsilon \rightarrow 0 \quad (23)$$

assim,

$$\|A_{\lambda(\epsilon)} t_\epsilon - M^\dagger t\| \leq \|A_{\lambda(\epsilon)} t_\epsilon - A_{\lambda(\epsilon)} t\| + \|A_{\lambda(\epsilon)} t - M^\dagger t\| \leq \|A_{\lambda(\epsilon)}\| \|t_\epsilon - t\| + \|A_{\lambda(\epsilon)} t - M^\dagger t\| \rightarrow 0 \quad (24)$$

Logo, se  $t_\epsilon$  está próximo do tempo de percurso  $t$  então  $A_{\lambda(\epsilon)} t_\epsilon$  está próximo da solução aproximada  $M^\dagger t$ .

Como exemplo desse método consideremos a parada em um processo iterativo (Gabriela Félix Brião, 2005). Seja então,

$$s^{(k+1)} = B_k s^{(k)} + C_k t \quad (25)$$

um processo iterativo e assuma que  $s^{(k)} \rightarrow M^\dagger t$ . Para cada  $\lambda > 0$ , seja  $k(\lambda)$  índice tal que  $k(\lambda) \rightarrow \infty$  quando  $\lambda \rightarrow 0$ . Então afirmamos que  $A_\lambda t = s^{(k(\lambda))}$  é uma regularização, pois

$$\lim_{\lambda \rightarrow 0} A_\lambda t = \lim_{\lambda \rightarrow 0} s^{(k(\lambda))} = M^\dagger t \quad (26)$$

por hipótese.

Frequentemente no caso da tomografia por tempo de percurso a matriz  $M$  (associada aos tempos de percurso dos raios sobre as células) tem posto deficiente ou é extremamente mal-condicionada. Isto leva naturalmente a necessidade de regularização.

### 3 Materiais e Métodos

#### 3.1 Área de Estudo

Para desenvolver a metodologia de tomografia sísmica para aquisição e processamento de dados em profundidades rasas (menores que 20 metros), inicialmente foi desenvolvido a técnica de maneira sintética e depois realizado uma aquisição em local com alvo conhecido, buscando otimizar os parâmetros de aquisição, e para calibrar as etapas de processamento. Tendo isso em vista, um alvo com dimensões e geometria conhecida foi instalado na área nos fundos do Observatório Sismológico no Campus Darcy Ribeiro da Universidade de Brasília (UnB), como pode ser observado na Figura 11 (página 29). Como resultado, são gerados mapas de anomalias de velocidade que representa o alvo enterrado, e com isso, estabelecer um esquema de aquisição que permita resolver o alvo.



Figura 11: Área de estudo nos fundos do Observatório Sismológico no Campus Darcy Ribeiro da Universidade de Brasília (UnB). A) Delimitando a área de estudo para a implantação do alvo. B) Imagem de satélite do Google Earth mostrando a localização da área de estudo (em vermelho).

### 3.1.1 Geologia

A região de estudo como um todo (UnB) encontra-se no Grupo Paranoá, de idade Meso/Neoproterozóico, composto majoritariamente por rochas sedimentares como quartzitos com intercalações de metassiltitos, calcários e dolomitos (Pimentel et al., 2011) e também uma parte no Grupo Bambuí que é a unidade sedimentar neoproterozóica mais importante do Brasil central. Ainda mais, possui uma sequência pelítica composta por folhelhos, argilitos e ritmitos finos de coloração verde que gradam para o topo da unidade para siltitos feldspáticos ou arcoseanos e que na porção centro-norte do Distrito Federal foram encontrados calcários e dolomitos micríticos, folhelhos, margas e siltitos argilosos e ricos em mica detritica atribuídos à base do Grupo Bambuí. Além deste, o Distrito Federal é composto geologicamente por outros dois conjuntos litológicos: grupos Canastra e Araxá. (Campos, 2004). (Figura 12, página 30).

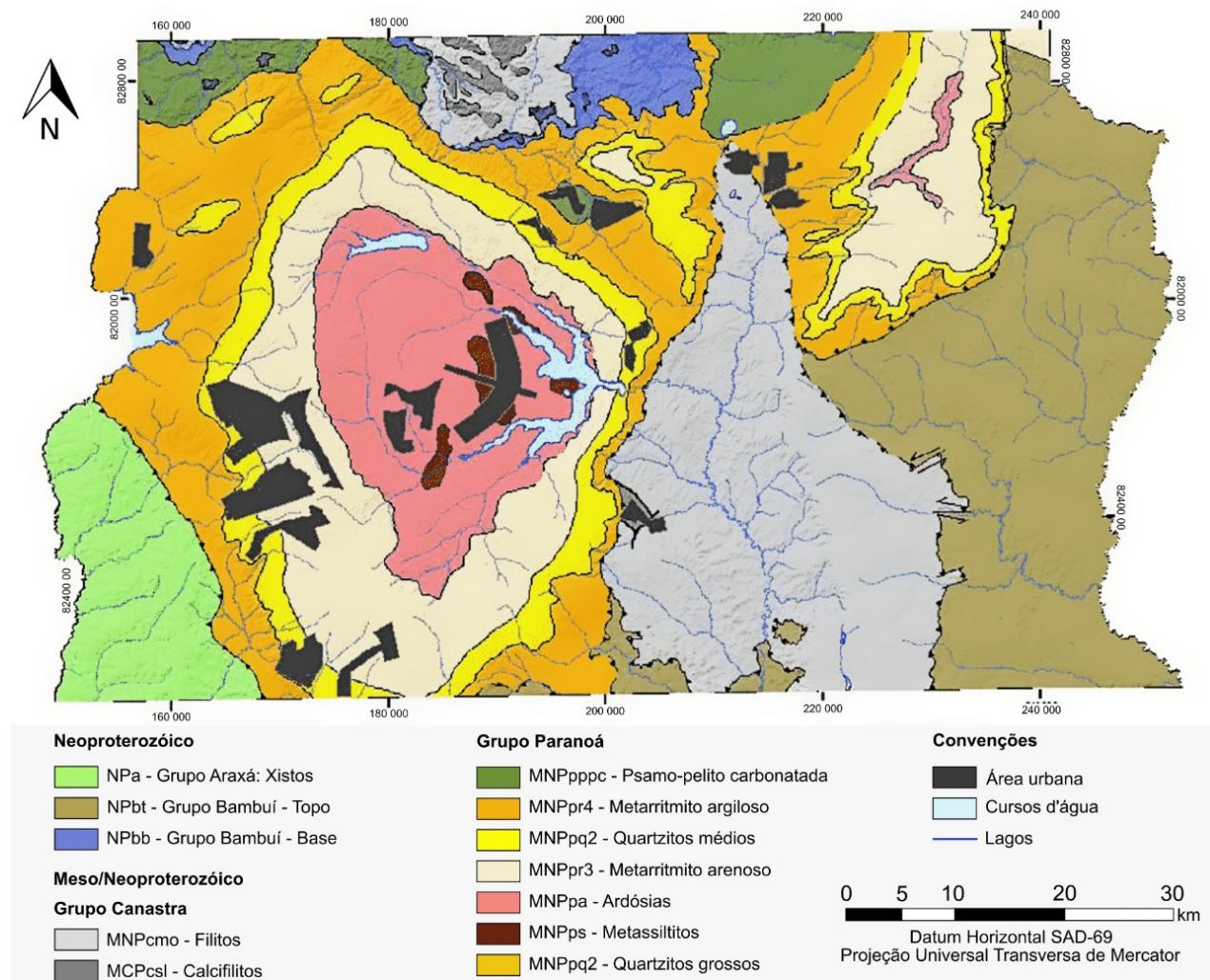


Figura 12: Mapa geológico do Distrito Federal. Modificado de Neumann (2012).

### 3.1.2 Pedologia

O solo da região de estudo é definido como Latossolo Vermelho. Com base no relatório da Empresa Brasileira de Pesquisa Agropecuária (FREITAS et al., 1978). O solo do DF pode ser classificado em dois núcleos em função dos tipos de coberturas. O primeiro é caracterizado pelo conjunto de tipos pedológicos mais abundantes, abrangendo 85% do território, e é constituído pelas seguintes classes de solos: Latossolo Vermelho, Latossolo Vermelho-Amarelo e Cambissolo Háplico. Já o segundo grupo, abrange aproximadamente 15% do território, e refere-se aos demais tipos de solos identificados dentro dos limites do DF, tais como: Nitossolo, Chernossolo, Gleissolo, Organossolo, Neossolo Quartzarênico, Neossolo Flúvico e Neossolo Litólico, além de Plintossolo. Assim também, o Plano Piloto de Brasília está localizado no primeiro grupo, com predominância de Latossolo Vermelho (Figura 13, página 32). Ainda mais, da área abrangida pelos latossolos no Distrito Federal (85%), o Latossolo Vermelho ocupa 38,63% (Martins, 2000). Então, na área de estudo, o Latossolo Vermelho ocupa 100% do solo.

Segundo o Zoneamento Ecológico-Econômico do Distrito Federal – ZEE-DF (2012), de maneira geral, os Latossolos Vermelhos possuem grande ocorrência associada à vegetação de cerrado e topos das chapadas. O material de origem é bastante variado, desde arenitos até rochas pelíticas, desde que possuam teores razoáveis de ferro. Normalmente, os Latossolos Vermelhos exibem-se com perfis profundos, muito porosos e bastante permeáveis. Ocorre nas chapadas mais elevadas e divisores de drenagem mais contínuos, sobre as rochas do Grupo Paranoá (Martins, 2000; ZEE-DF, 2012).

Este solo é constituído por uma sequência morfológica de horizontes com *A* moderado, *B* latossólico e *C*. O horizonte superficial *A* desenvolve-se com espessuras entre 20 e 50 cm e com coloração vermelho escuro. Neste horizonte há presença abundante de raízes, exibindo uma porção mais próxima a superfície, de tonalidade mais escura que a superior, indicando menor presença de matéria orgânica no nível superior por ação antrópica relativa a aterrramento, visto que a área de estudo é uma área de jardinagem e que houve construção civil. Em geral, o horizonte *A* tem estrutura granular, sendo muito friável quando úmido (Martins, 2000; ZEE-DF, 2012).

O horizonte subsuperficial *B* exibe um importante estágio de intemperização com textura argilosa e estrutura granular fraca, com espessura quase sempre maior que 250 cm. Pode ser subdividido nos sub- horizontes *Bw1* e *Bw2*, no entanto, possui pouca ou nenhuma diferenciação entre eles (ZEE-DF, 2012).

O horizonte *C* caracteriza-se como a camada abaixo do *solum* (horizontes *A* e *B*) menos afetada por processos pedogenéticos mantendo características da rocha original. É conhecido ainda como manto de alteração ou saprolito.

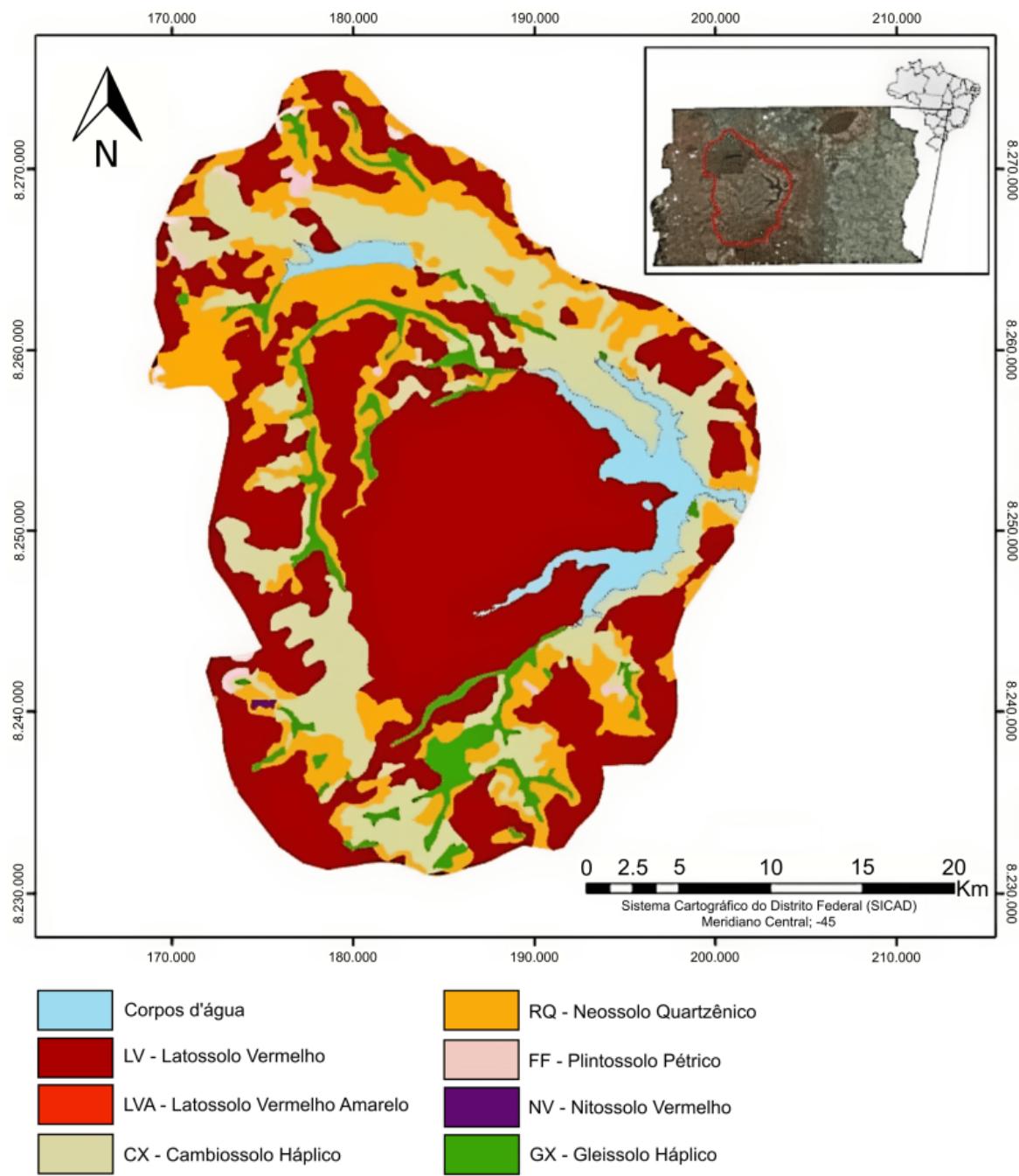


Figura 13: Mapa de solos da Bacia do Lago Paranoá. Modificado de Menezes (2010).

## 3.2 Frente teórica-computacional

O trabalho foi elaborado em duas frentes: Uma teórico-computacional e outra experimental. Na frente teórico-computacional realizamos o desenvolvimento de rotinas computacionais para inversão dos dados de tempo de percurso (utilizando o tempo de chegada da onda direta). Utilizou-se de conceitos de inversão por mínimos quadrados com algum nível de regularização para controlar a ambiguidade do modelo de parâmetros (velocidade/vagarosidade). A estrutura computacional se necessária seria fornecida pelo Observatório Sismológico da UnB, e os programas/linguagens de programação utilizados são de código aberto (como Python, Matlab, Shellscript, etc). Elaborou-se modelos sintéticos com posição e propriedades físicas conhecidas das estruturas a serem simuladas. Os ajustes que por ventura sejam necessários foram incluídas no problema inverso, ou ainda, no esquema de aquisição.

### 3.2.1 Recursos computacionais

Para a construção da rotina computacional, tanto para o desenvolvimento da etapa de gerar dados sintéticos com também para a etapa de processar os dados de campo para obter os mapas de tomografia sísmica, usamos a linguagem de programação Python. Ainda mais, é uma linguagem de programação de alto nível, ou seja, com sintaxe mais simplificada e próxima da linguagem humana, utilizada nas mais diversas aplicações, como desktop, web, servidores e ciência de dados e mais especificadamente com a biblioteca pyGIMLI, a utilização dessa linguagem atualmente encontra-se em crescimento no ramo da Geofísica no que diz respeito ao desenvolvimento de diversas rotinas de processamentos, utilizando-se dos conceitos dos métodos Geofísicos.

Por outro lado, pyGIMLI é uma biblioteca de código aberto para modelagem e inversão em Geofísica(Rücker et al., 2017). A biblioteca orientada a objetos fornece gerenciamento para malhas estruturadas e não estruturadas em 2D e 3D. Uma tarefa principal do pyGIMLI é realizar inversão. Vários tipos de regularização em malhas (1D, 2D, 3D) com disposição regular ou irregular estão disponíveis. Existe um controle flexível de todos os parâmetros de inversão. A estrutura de inversão padrão é baseada no método generalizado de Gauss-Newton. O pyGIMLI vem com vários operadores Geofísicos avançados, que podem ser usados diretamente para um determinado problema.

Então, para que fosse possível o desenvolvimento do *software*, as principais bibliotecas usadas foram o Matplotlib, Numpy e o pyGIMLI. O Matplotlib é um pacote de gráficos 2D usado em Python para desenvolvimento de aplicativos, scripts interativos e geração de imagens com qualidade de publicação em interfaces de usuário e sistemas operacionais (ver mais em Hunter, 2007). No mundo Python, o NumPy é o pacote fundamental para computação científica. É uma biblioteca Python que fornece um objeto array multidimensional, vários objetos derivados (como arrays e matrizes) e uma variedade de rotinas para operações rápidas em arrays, incluindo matemática, lógica, manipulação de formas, classificação, seleção, I/O , transformadas discretas de Fourier, álgebra linear básica, operações estatísticas básicas e simulação aleatória (ver mais em <https://numpy.org/doc/1.22/>).

### **3.2.2 pyGIMLi: Uma biblioteca de código aberto para modelagem e inversão em Geofísica**

O pyGIMLi oferece uma vantagem distinta, tal que pode ser facilmente estendido por módulos compilados de C ou Fortran, por exemplo, permitindo aos usuários estender o código licenciado ou terceirizar partes demoradas em extensões computacionalmente eficientes. Realiza-se o uso dessa flexibilidade e implementa-se todas as partes sensíveis ao tempo de execução em uma biblioteca principal C++. As ligações completas do Python para essa biblioteca principal são complementadas por funcionalidades escritas em Python puro, oferecendo, assim, eficiência e flexibilidade para o rápido desenvolvimento de aplicativos robustos de modelagem e inversão. O componente de modelagem oferece gerenciamento de malha, bem como solucionadores de elementos finitos e volumes finitos em 1D, 2D e 3D. O componente de inversão é baseado em um algoritmo determinístico de Gauss-Newton e funciona com qualquer operador direto físico fornecido. Várias rotinas de pós-processamento são fornecidas para visualizar os resultados em 2D usando Matplotlib (Rücker et al., 2017) e em 3D usando o software ParaView (Ayachit et al., 2015) ou Mayavi (Ramachandran e Varoquaux, 2011).

Um dos principais benefícios do uso do Python é a prototipagem abstrata, por exemplo, é implementado o principal solucionador inverso em C++ com o uso de uma matriz base abstrata e com isso é possível usá-lo com qualquer matriz avançada personalizada diretamente do Python. Isso resulta em flexibilidade dos tipos de matriz usados com esforço mínimo de codificação e deficiências mínimas de tempo de execução. Além de tipos densos e diferentes de matrizes esparsas, existem matrizes especializadas como matrizes em escala de linha ou coluna ou matrizes de Kronecker. Além disso, é implementada uma matriz de blocos contendo referências a um número de matrizes de tipo arbitrário. Ele pode ser usado para matrizes de restrição eficientes ou matrizes Jacobianas conjuntas sem perda de desempenho.

#### **pyGIMLi |Estruturas de inversão**

Os *frameworks* de inversão são abordagens generalizadas e abstratas para resolver um problema específico de inversão sem especificar os métodos geofísicos apropriados. Isso pode ser uma estratégia de regularização específica, uma formulação alternativa do problema inverso ou algoritmos de inversão de rotina.

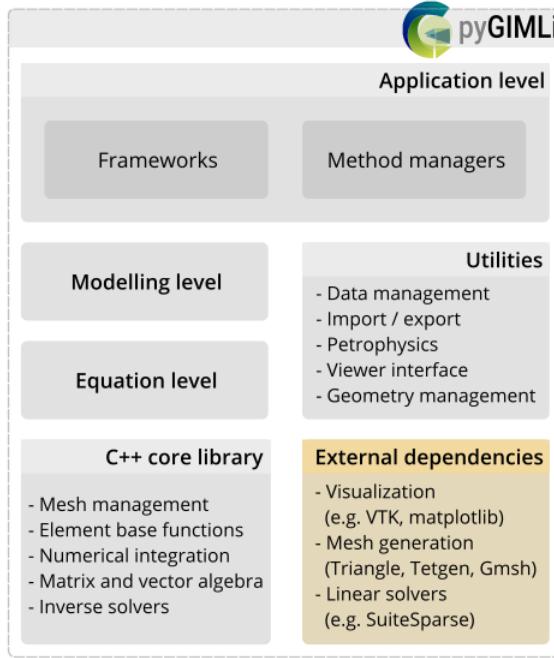


Figura 14: Arquitetura de software do pyGIMLi ilustrando seus componentes e diferentes níveis de abstração. A biblioteca Python foi construída sobre um núcleo C++ e várias dependências externas (caixa amarela). (Para interpretação das referências à cor nesta legenda da figura, o leitor deve consultar a versão web <https://www.pygimli.org/design.html#sec-design>).

- *Inversão de Gauss-Newton*

A estrutura de inversão padrão é baseada no método generalizado de Gauss Newton e é compatível com qualquer operador direto e, portanto, aplicável a vários problemas físicos. Afirmamos o problema de inversão como minimização de uma função objetivo que consiste em desajuste de dados e restrições de modelo:

$$\|\mathbf{W}_d(\mathcal{F}(\mathbf{m}) - \mathbf{d})\|_2^2 + \lambda \|\mathbf{W}_m(\mathbf{m} - \mathbf{m}_0)\|_2^2 \rightarrow \min \quad (27)$$

Observe que não foi incluído restrições de desigualdade na minimização, mas são usadas transformações para restringir os parâmetros a intervalos razoáveis (por exemplo, Kim e Kim, 2011).  $\mathbf{W}_d$  é a matriz de ponderação de dados contendo os erros de dados inversos,  $\mathbf{W}_m$  é a matriz de restrição do modelo (por exemplo, um operador de rugosidade de primeira ordem) e  $\mathbf{m}_0$  é um modelo de referência. O fator adimensional  $\lambda$  dimensiona a influência do termo de regularização. Existe uma ampla gama de diferentes métodos de regularização (diferentes tipos de suavidade e amortecimento, operadores mistos, suavização anisotrópica). A aplicação do esquema de Gauss-Newton na minimização (Equação 27 da página 35) produz a atualização do modelo  $\Delta\mathbf{m}^k$  na  $k$ -ésima iteração (Park e Van, 1991):

$$\begin{aligned}
 (\mathbf{J}^T \mathbf{W}_d^T \mathbf{W}_d \mathbf{J} + \lambda \mathbf{W}_m^T \mathbf{W}_m) \Delta\mathbf{m}^k &= \mathbf{J}^T \mathbf{W}_d^T \mathbf{W}_d (\Delta\mathbf{d}^k) - \lambda \mathbf{W}_m^T \mathbf{W}_m (\mathbf{m}^k - \mathbf{m}^0) \\
 \text{com } \Delta\mathbf{d}^k &= \mathbf{d} - \mathcal{F}(\mathbf{m}^k) \\
 \text{e } \Delta\mathbf{m}^k &= \mathbf{m}^k - \mathbf{m}^{k-1}
 \end{aligned} \quad (28)$$

que é resolvido usando um solucionador de mínimos quadrados de gradiente conjugado (Günther et al., 2006). O processo de inversão incluindo a regularização específica da região é esboçado na Figura 15 (página 36).

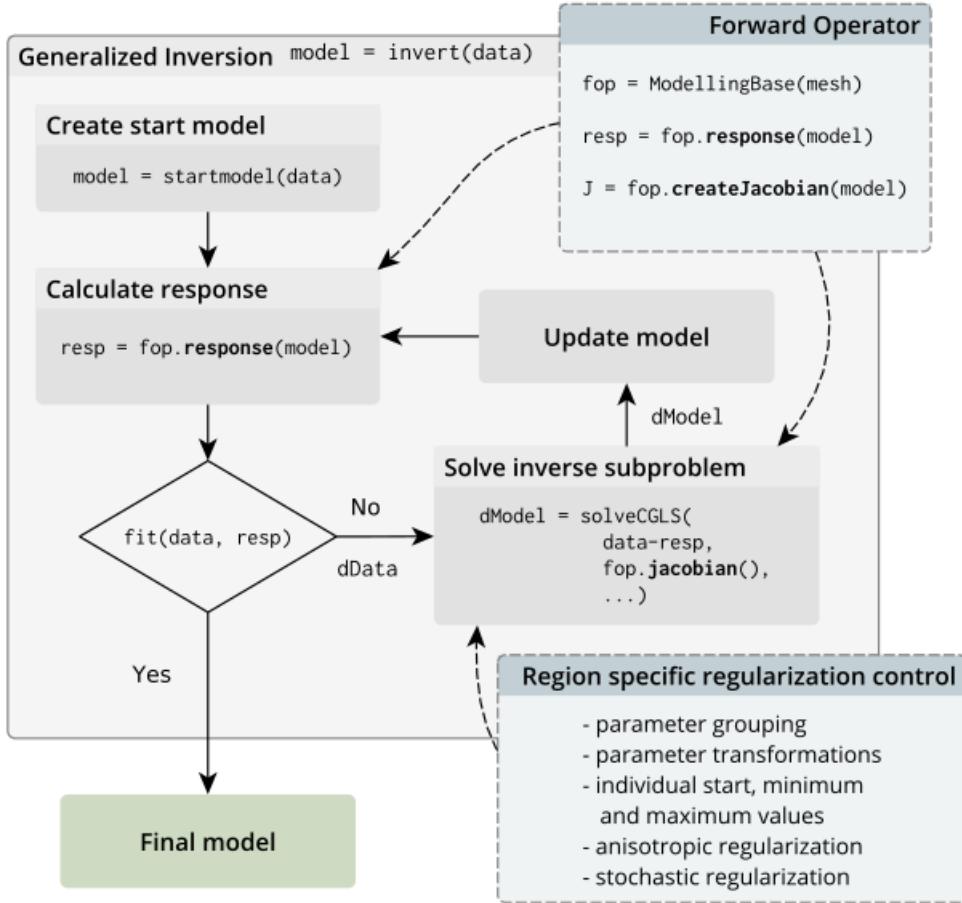


Figura 15: Esquema de inversão generalizada. Operadores de encaminhamento já implementados ou personalizados podem ser usados para fornecer a função de resposta específica do problema e sua Jacobiana. Várias estratégias estão disponíveis para regularizar o problema inverso.

Todas as matrizes da formulação de inversão podem ser acessadas diretamente do Python e, assim, oferecem oportunidades para análise de incerteza e resolução, bem como design experimental (por exemplo, Wagner et al., 2015). Além de diferentes abordagens de inversão, existem os chamados *frameworks* para tarefas típicas de inversão (principalmente de regularização). Exemplos que já estão implementados no pyGIMLi são, por exemplo:

**Marquardt scheme** inversão de alguns parâmetros independentes, por exemplo, ajuste de espectros (Loewer et al., 2015).

**Soil-physical model reduction** incorporando funções físicas do solo (Igel et al., 2016; Costabel e Günther, 2014).

**Classical joint inversion** de dois conjuntos de dados para o mesmo parâmetro como DC e EM (Günther, 2013).

**Block joint inversion** de vários dados 1D usando camadas comuns, por exemplo, MRS+VES (Günther e Müller-Petke, 2012).

**Sequential (constrained) inversion** inversão independente sucessiva de conjuntos de dados, por exemplo, *classic time-lapse inversion* (por exemplo, Hübner et al., 2015).

**Simultaneous constrained inversion** de conjuntos de dados de dados vizinhos no espaço (LCI, por exemplo, Costabel et al., 2016), tempo (*full time-lapse*) ou frequência (Günther e Martin, 2016).

**Structurally coupled cooperative inversion** de dados díspares com base na semelhança estrutural (por exemplo, Ronczka et al., 2017).

**Structure-based inversion** usando modelos 2D em camadas (Attwa et al., 2014).

## pyGIMLi |Parametrização

As malhas 2-D utilizadas para as rotinas de modelagem direta são compostas de retângulos aos quais são atribuídas velocidades, densidades e resistividades constantes. Os tamanhos das células são adaptados individualmente para cada método para levar em conta problemas de precisão numérica e eficiência computacional.

## pyGIMLi |Modelagem direta

Para tomografia sísmica, os tempos de primeira chegada são calculados por um *eikonal solver* (Podvin e Lecomte, 1991) e depois os caminhos de raios associados são construídos por um *steepest descent method* (Aldridge et al., 1993).

## pyGIMLi |Meshing

### • Unstructured Meshes

Um passo essencial na simulação numérica é encontrar uma discretização adequada de um domínio contínuo. Este é o problema da geração de malha. A geração de malhas é um componente chave para a simulação computacional de problemas físicos e de engenharia onde são usados métodos numéricos como os métodos dos elementos finitos, das diferenças finitas e dos volumes finitos (Teng e Wong, 2000). Um método de simulação numérica normalmente aplica as seis etapas básicas a seguir.

1. **Modelagem matemática:** definir o domínio contínuo  $\Omega$  e as equações diferenciais parciais (EDP) sobre o domínio que modelam com precisão o problema físico e de engenharia;
2. **Modelagem geométrica:** aproximar o domínio contínuo  $\Omega$  com uma descrição discreta;
3. **Geração de malha:** decompor o interior do domínio em uma malha  $M$  de elementos simples e "bem modelados", como caixas ou *simplices* com uma proporção limitada;
4. **Aproximação numérica:** construir um sistema de equações lineares ou não lineares sobre  $M$  para as EDPs governantes;

5. **Solução numérica:** resolver o sistema de equações e estimar o erro da solução;
6. **Refinamento adaptativo:** se necessário, refine a malha e repita os passos 5 e 6 sobre as malhas refinadas.

Uma vez que tenhamos gerado uma malha, as equações diferenciais para fluxo, ondas e distribuição de calor são então aproximadas pelas formulações de diferença finita, elemento ou volume. No entanto, nem todas as malhas são igualmente boas numericamente. Os erros de discretização dependem da forma geométrica e do tamanho dos elementos, enquanto a complexidade computacional para encontrar a solução numérica depende do número de elementos na malha e também da qualidade geométrica geral da malha; o objetivo é gerar malhas pequenas e numericamente sólidas.

A forma mais simples de uma malha é uma malha estruturada. Existem dois tipos de malhas estruturadas: malhas *geometricamente estruturadas* e *topologicamente estruturadas*. Exemplos de malhas geometricamente estruturadas são grades cartesianas regulares e grades hexagonais uniformes. Nessas malhas, todos os elementos são geometricamente iguais. Uma malha é estruturada topologicamente se sua estrutura topológica for isomórfica à de uma malha estruturada geometricamente. Por exemplo, podemos aplicar uma transformação conforme a uma grade estruturada para gerar uma malha estruturada topologicamente. As grades estruturadas são fáceis de gerar e manipular, o que facilita o uso de estruturas de dados simples para reduzir a complexidade da programação. Além disso, a teoria numérica sobre esses tipos de discretização é bem compreendida. No entanto, o uso de malhas regulares estruturadas limita a aplicabilidade dos métodos numéricos a problemas cujos domínios são simples e cujas funções de solução são suaves.

Para problemas com limites geométricos complexos e com soluções que mudam rapidamente, precisamos usar uma malha não estruturada que tenha topologia e espaçamento local variáveis para reduzir o tamanho do problema. Para o exemplo de modelagem de terremotos, precisamos de uma discretização muito mais densa e mais fina perto do centro do terremoto, embora seja desejável não desperdiçar pontos de malha em regiões com baixa atividade. A adaptabilidade de malhas não estruturadas traz novos desafios, especialmente para problemas 3D; a teoria numérica torna-se mais difícil e o projeto algorítmico torna-se mais difícil.

- *Unstructured Triangular Meshes*

A malha mais geral e versátil é uma malha triangular não estruturada na qual cada elemento é um simplex, ou seja, um triângulo em 2D ou um tetraedro em 3D. Em geral, um *k-simplex* é um politopo convexo de  $k + 1$  pontos de dimensão  $k$ . Uma malha triangular é uma triangulação do domínio de entrada (por exemplo, um polígono), juntamente com alguns pontos extras, chamados de *Steiner points*. Uma *triangulação* ou um *complexo simplicial* é uma decomposição de um domínio em uma coleção de *interior disjointed simplices*, de modo que dois *simplices* só podem se cruzar em um simplex de dimensão inferior, ou seja, elementos vizinhos são conformes em seus limites. Combinatoriamente, uma triangulação  $T$  pode ser expressa como um *Piecewise Linear System* (PLS) de um conjunto  $S_T$  de *simplices*: se  $s$  é um simplex em  $S_T$  então todas as suas faces de menor dimensão, que também são *simplices*, também pertencem a  $S_T$  (ver mais em Teng e Wong, 2000).

Para aproximar adequadamente uma função contínua, além da condição de que uma malha deve estar de acordo com os limites do domínio e ser suficientemente fina, cada elemento individual da malha deve ser bem modelado. Um critério de forma comum para elementos é a condição de que os ângulos de cada elemento não sejam nem muito pequenos nem muito grandes, ou que a proporção de cada elemento seja limitada.

Diversas definições de razão de aspecto têm sido utilizadas na literatura, e se uma delas é limitada por uma constante, todas também são. Por exemplo, em 2D, pode-se usar o menor ângulo de um triângulo para medir sua condição de forma. Em 3D, pode-se usar o menor ângulo diedro de um tetraedro. A seguinte definição de razão de aspecto, popularizada por Mitchell e Vavasis, é definida uniformemente para qualquer dimensão.

- *Structured grids*

Uma malha cartesiana é facilmente gerada com algumas linhas simples de codificação de computador, em oposição à construção manual demorada de malhas não estruturadas triangulares/tetraédricas. Associado a uma grade de fundo cartesiana está um número arbitrário de elementos de origem prescritos pelo usuário nos quais os parâmetros da grade são definidos. Os elementos de origem propagam os parâmetros de espaçamento no campo de forma automática e sistemática. Existem dois tipos de elementos: elementos nodais e lineares. As fontes podem ser posicionadas em qualquer lugar no campo, preferencialmente perto das superfícies da geometria e nos limites externos. De um modo geral, a localização de um elemento fonte é escolhida onde uma distribuição bem controlada de pontos de grade é desejada em uma grade de baixa qualidade e em alguns casos afeta severamente o processo de *front advancement* (ver mais em Pirzadeh, 1993).

- *Métodos Refinement*

A técnica mais popular para geração de conjuntos de pontos e elementos é o Refinamento Iterativo e Adaptativo: Primeiro, uma triangulação inicial é construída. *Steiner points* adicionais são então inseridos no domínio e uma triangulação melhorada é gerada. Mais *Steiner points* são então adicionados para uma melhor triangulação, e assim por diante até que uma malha de qualidade seja obtida (ver mais em Teng e Wong, 2000).

- *Tomografia de tempos de percurso (Traveltimes tomography)*

Esquemas de tomografia baseados em raios usam a aproximação de raios para descrever a propagação da onda através da região de estudo, geralmente chamada de *raytracing*. Vários métodos foram desenvolvidos para realizar o *raytracing*, seguindo diferentes abordagens teóricas. Para tomografia de tempo de percurso, o problema inverso é resolvido por tempos de percurso e caminhos de raios, e o traçado de raios deve ser realizado várias vezes de forma iterativa (Giroux e Larouche, 2013).

O método *Shortest Path* (Moser, 1991) com nós secundários (Giroux e Larouche, 2013) é o algoritmo subjacente implementado no pyGIMLi. Essa abordagem procura o caminho mais rápido através de uma determinada malha da fonte ao receptor. O problema discretizado é um sistema linear de equações que os métodos numéricos podem resolver.

De acordo com Ronczka et al. (2017), o problema linearizado é dado por  $\vec{\Delta d} = G\vec{\Delta m}$ , onde a matriz Jacobiana  $G$  contém as derivadas parciais  $\partial t_i / \partial m_j$  e os tempos de deslocamento observados são mantidos em  $\vec{d}$ . A tomografia de refração sísmica é um problema inverso que pode ser resolvido por uma minimização restrita de suavidade da função de custo dada por

$$\Phi = \Phi_d + \lambda \Phi_m = \sum_{i=1}^N \left( \frac{d_i - f_i(m)}{\epsilon_i} \right)^2 + \lambda \|Cm\|_2^2 \quad (29)$$

Onde  $\Phi_d$  é o desajuste dos dados ponderados pelo erro,  $\Phi_m$  é a rugosidade do modelo e  $\lambda$  é o parâmetro de regularização. O tempo de percurso  $t$  entre a fonte e o receptor ao longo de um caminho de raio é calculado por

$$t = \sum_{i=1}^n \frac{r_i}{v_i} \quad (30)$$

Onde  $r_i$  e  $1/v_i$  são o comprimento do caminho (percurso) e a vagarosidade para um segmento  $i$ , respectivamente. A diferença entre os pontos de dados individuais  $d_i$  e os correspondentes tempos de percurso teóricos  $f_i(m)$  é ponderada pelos seus erros individuais  $\epsilon_i$ .

Um modelo ponderado funcional  $W_c$  pode ser usado para incorporar restrições de modelo adicionais na função objeto estendendo a rugosidade  $\Phi_m$ , resultando em

$$\Phi_m = \|W_c Cm\|_2^2 \quad (31)$$

A rugosidade consiste na matriz derivada  $C$  aplicada ao modelo  $m$ . Cada linha em  $C$  está associada a um limite, onde a matriz de ponderação  $W_c$  é uma matriz diagonal onde cada  $w_i$  representa fatores de penalidade para os diferentes limites das células do modelo.

### 3.2.3 Estrutura do *Software*

Todo o código foi escrito em Python (versão 3.8.10) utilizando a distribuição Anaconda (que é uma plataforma de distribuição Python de código aberto). Nessa plataforma é possível criar um ambiente virtual e instalar todas as dependências necessárias para a criação do *software*. Então foi criado um ambiente virtual, isso é feito porque é uma prática recomendada no mundo da programação para evitar conflitos com outras dependências. A partir disso, usou-se o Visual Studio Code (VS Code) que é um editor de código de código aberto desenvolvido pela Microsoft e que encontra-se presente nessa plataforma.

Os testes foram realizados em sistemas Windows 10 e Ubuntu 20.04.3 LTS (ambos de 64 bits). O InTTraPy (Invert Travel-Time Python) é dividido em dois módulos: InTTraPy sintético, para realização de estudos de casos sintéticos, e InTTraPy campo, para processar os dados adquiridos em campo. Um fluxo geral de operação sobre como usar o InTTraPy é mostrado na Figura 16 (página 43).

## InTTraPy | Sintéticos

No desenvolvimento do módulo InTTraPy Sintético, começamos importando os pacotes necessários (*matplotlib.pyplot*, *numpy*, *pygimli*, *pygimli.meshTools* e *pygimli.physics.traveltime*). Em seguida, foi desenvolvida a etapa que permite a construção da geometria de aquisição (permitindo estabelecer as posições dos geofones e um esquema de medição, que consiste em índices de fonte e receptor). Depois usamos o gerenciador *TravelTime* e atribuímos velocidades de onda P. Para isso, criamos um mapa dos marcadores de células de 0 a 3 para velocidades (em m/s) e geramos um vetor de velocidade. Os cálculos de tempo de percurso funcionam em malhas não estruturadas e grades estruturadas. Logo, os dados sintéticos são simulados em uma malha não estruturada e invertidos em uma grade estruturada simples.

Posteriormente, criamos um *DataContainer* (para armazenar, carregar e salvar dados no formato de dados unificado GIMLi. Por padrão, os dados com índices predefinidos como ‘s’, ‘g’ e ‘t’ são *tokens* para números de fontes, geofones e tempo de percurso da onda direta.) vazio e o preenchemos com as posições dos sensores, todos os pares de fontes possíveis e seus tempos de percurso para o cenário que o usuário desejar simular.

A simulação direta é realizada com algumas linhas de código. Inicializamos uma instância do *Refraction manager* e chamamos sua função *simulate* com a malha, o esquema e o modelo de vagarosidade ( $1/v$ ). Também adicionamos 0,1% *relative* e 10 microssegundos de ruído absoluto. Os nós secundários permitem simulações diretas e mais precisas (ver mais em Giroux e Larouche, 2013). Logo depois, é criada uma grade estruturada como malha de inversão com a geometria informada pelo usuário. Configuramos também *setRecalcJacobian(False)* para *setRecalcJacobian(True)* para simular a inversão linear.

Por fim, é gerado o modelo e o resultado da inversão um ao lado do outro. Também é gerado um mapa do caminho que os raios percorreram. Todas as telas de plotagem podem ser exportadas como imagens de alta resolução.

## InTTraPy | Campo

Para o desenvolvimento do módulo InTTraPy Campo, utilizou-se as mesmas etapas de desenvolvimento do sintético adaptando-o para receber os dados de campo (arquivos *nome.vs*) com as informações dos tempos de chegada da onda direta, bem como, as informações dos parâmetros utilizados na aquisição.

Tendo isso em vista, o *software* possui no momento, somente a parte do *Back End* (a lógica de programação), a parte do *Front-End* (parte visual com a qual os usuários podem interagir) será desenvolvida em um momento posterior. Mas o funcionamento do *software* para o usuário funciona da seguinte forma: assim que o usuário inicializar o *software*, é carregado uma tela com as opções de escolher o módulo do InTTraPy Sintético ou Campo.

Para o InTTraPy Sintético, o usuário entra com os parâmetros da Tabela 1 (página 42). Após informar os parâmetros, é gerado os dados com as informações das posições dos Geofones e Fontes que são salvas posteriormente no *DataContainer*. Em seguida é feito o processamento e o usuário tem a opção de salvar os dados com as posições dos Geofones e Fontes em um arquivo *JSON* (*JavaScript Object Notation*). Em seguida, os

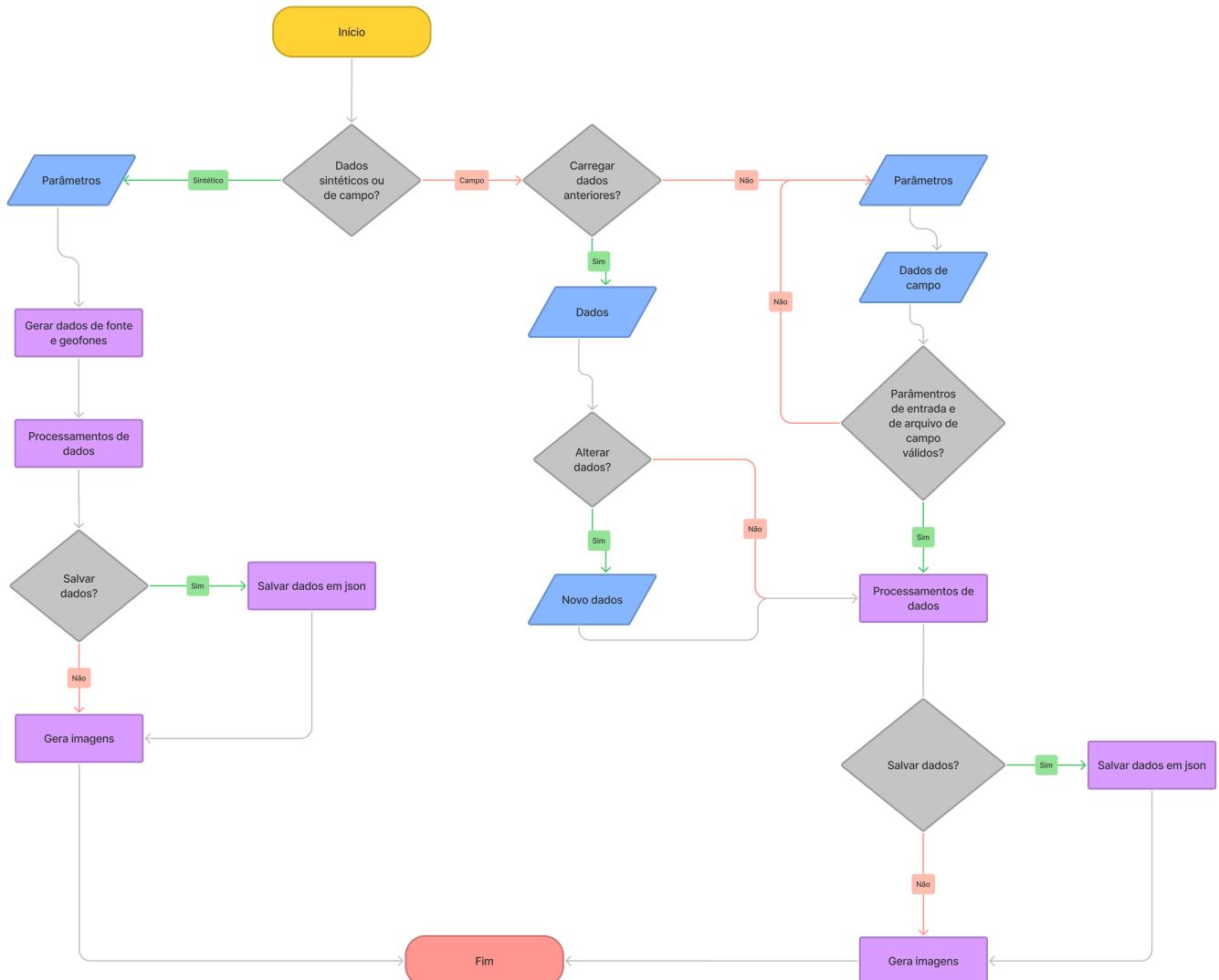
modelos sintéticos de tomografia sísmica por tempo de percurso é gerado e os mapas podem ser exportados e a execução do *software* pode ser finalizado.

Tabela 1: Parâmetros que devem ser informados para gerar os modelos sintéticos. Observação: posteriormente, será feito uma atualização para que o usuário possa modelar outras geometrias para alvo.

Parâmetros
Dimensão da área de estudo no eixo x (metros)
Dimensão da área de estudo no eixo y (metros)
Dimensão do alvo no eixo x (metros)
Dimensão do alvo no eixo y (metros)
Espaçamento dos Geofones no eixo x (metros)
Espaçamento dos Geofones no eixo y (metros)
Espaçamento das Fontes no eixo x (metros)
Espaçamento das Fontes no eixo y (metros)
Posição do primeiro Geofone no eixo x (metros)
Posição do primeiro Geofone no eixo y (metros)
Posição da primeira Fonte no eixo x (metros)
Posição da primeira Fonte no eixo y (metros)
Número de Geofones no eixo x
Número de Geofones no eixo y
Número de Fontes no eixo x
Número de Fontes no eixo y
Refinamento para a grade de inversão

Para o InTTraPy Campo, o usuário tem a opção de carregar dados anteriores (ou seja, arquivos *JSON* com as posições dos Geofones, Fontes e tempos de percurso). Ao carregar o arquivo, o usuário pode optar por modificar alguma informação referentes as posições de Geofones e/ou Fontes. Logo depois, os dados são processados e o usuário tem a opção de salvar os dados em *JSON*. Logo após, os modelos reais de tomografia sísmica por tempo de percurso é gerado e os mapas podem ser exportados e a execução do *software* pode ser finalizado.

Por outro lado, o usuário optando por não carregar dados anteriores, deve-se informar o parâmetros Tabela 2 (página 44).



#### Legenda



Figura 16: Fluxo de operação geral com as etapas usadas em cada módulo: Geração de modelos sintéticos de tomografia sísmica (InTTraPy Sintético) e processamento de dados de tempo de percurso gerando modelos reais de tomografia sísmica (InTTraPy Campo).

Tabela 2: Parâmetros que devem ser informados para gerar os modelos reais.

Parâmetros
Dimensão da área de estudo no eixo x (metros)
Dimensão da área de estudo no eixo y (metros)
Espaçamento dos Geofones no eixo x (metros)
Espaçamento dos Geofones no eixo y (metros)
Espaçamento das Fontes no eixo x (metros)
Espaçamento das Fontes no eixo y (metros)
Posição do primeiro Geofone no eixo x (metros)
Posição do primeiro Geofone no eixo y (metros)
Posição da primeira Fonte no eixo x (metros)
Posição da primeira Fonte no eixo y (metros)
Número de Geofones no eixo x
Número de Geofones no eixo y
Número de Fontes no eixo x
Número de Fontes no eixo y
Refinamento para a grade de inversão

### 3.3 Etapa experimental - Aquisição em local com alvo conhecido

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris

lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

## 4 Resultados e Discussão

### 4.1 Dados sintéticos - Estudos de casos

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

## 4.2 Dados de campo - Processamento e inversão dos dados sísmicos

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

## 5 Conclusões

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

## Referências Bibliográficas

- Aldridge, D. F., D. W. Oldenburg, et al., 1993, Two-dimensional tomographic inversion with finite-difference traveltimes: *Journal of Seismic Exploration*, **2**, 257–274.
- Attwa, M., I. Akca, A. T. Basokur, e T. Günther, 2014, Structure-based geoelectrical models derived from genetic algorithms: a case study for hydrogeological investigations along elbe river coastal area, germany: *Journal of Applied Geophysics*, **103**, 57–70.
- Axelsson, O., 1994, Iterative solution methods.: Cambridge University Press.
- Ayachit, U., A. Bauer, B. Geveci, P. O’Leary, K. Moreland, N. Fabian, e J. Mauldin, 2015, Paraview catalyst: Enabling in situ data analysis and visualization: *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization*, 25–29.
- Bording, R. P., A. Gersztenkorn, L. R. Lines, J. A. Scales, e S. Treitel, 1987, Applications of seismic travel-time tomography: *Geophysical Journal International*, **90**, 285–303.
- Branham, R. L., 1990, Introduction to overdetermined systems, *in* *Scientific Data Analysis*: Springer, 67–83.
- Campos, J. E. G., 2004, Hidrogeologia do distrito federal: bases para a gestão dos recursos hídricos subterrâneos: *Brazilian Journal of Geology*, **34**, 41–48.
- Cao, S., e S. Greenhalgh, 1995, Relative-error-based non-linear inversion: Application to seismic traveltimes tomography: *Geophysical Journal International*, **121**, 684–694.
- Computations, M., 1989, Johns hopkins univ: Press, Baltimore, MD.
- Costabel, S., e T. Günther, 2014, Noninvasive estimation of water retention parameters by observing the capillary fringe with magnetic resonance sounding: *Vadose Zone Journal*, **13**.
- Costabel, S., T. Günther, R. Dlugosch, e M. Müller-Petke, 2016, Torus-nuclear magnetic resonance: Quasicontinuous airborne magnetic resonance profiling by using a helium-filled balloon: *Geophysics*, **81**, WB119–WB129.
- Fernandez, A. L., 2000, Tomographic imaging the state of stress: Georgia Institute of Technology.
- FREITAS, F. d., I. GOMES, R. FERREIRA, e L. ANTONELLO, 1978, Levantamento de reconhecimento dos solos do distrito federal: Embrapa-Serviço Nacional de Levantamento e Classificação de Solos, Rio de Janeiro.
- Gabriela Félix Brião, P. D. J. P. Z., 2005, Tomografia sísmica por tempo de percurso: Modelagem, métodos numéricos e implementação.
- Giroux, B., e B. Larouche, 2013, Task-parallel implementation of 3d shortest path ray-tracing for geophysical applications: *Computers & geosciences*, **54**, 130–141.
- Günther, T., 2013, On inversion of frequency domain electromagnetic data in salt water problems-sensitivity and resolution: *Near Surface Geoscience 2013-19th EAGE European Meeting of Environmental and Engineering Geophysics*, European Association of Geoscientists & Engineers, cp-354.
- Günther, T., e T. Martin, 2016, Spectral two-dimensional inversion of frequency-domain induced polarization data from a mining slag heap: *Journal of Applied Geophysics*, **135**, 436–448.
- Günther, T., e M. Müller-Petke, 2012, Hydraulic properties at the north sea island of borkum derived from joint inversion of magnetic resonance and electrical resistivity soundings: *Hydrology and Earth System Sciences*, **16**, 3279–3291.
- Günther, T., C. Rücker, e K. Spitzer, 2006, Three-dimensional modelling and inversion of dc resistivity data incorporating topography—ii. inversion: *Geophysical Journal*

- International, **166**, 506–517.
- Hübner, R., K. Heller, T. Günther, e A. Kleber, 2015, Monitoring hillslope moisture dynamics with surface ert for enhancing spatial significance of hydrologic point measurements: *Hydrology and Earth System Sciences*, **19**, 225–240.
- Hunter, J. D., 2007, Matplotlib: A 2d graphics environment: *Computing in Science Engineering*, **9**, 90–95.
- Igel, J., S. Stadler, e T. Guenther, 2016, High-resolution investigation of the capillary transition zone and its influence on gpr signatures: 2016 16th International Conference on Ground Penetrating Radar (GPR), IEEE, 1–5.
- Imhof, A., 2007, Caracterización de arenas y gravas con ondas elásticas: Tomografía sísmica en cross hole: XIV First Prize Award Fundación García Siñérez for the Best Doctoral Thesis in Pure or Applied Geophysics.
- Imhof, A. L., e C. A. Calvo, 2018, A variational formulation to image inclusions in 2d travel time cross hole tomography: *Brazilian Journal of Geophysics*, **21**, 269–274.
- Kim, H. J., e Y. Kim, 2011, A unified transformation function for lower and upper bounding constraints on model parameters in electrical and electromagnetic inversion: *Journal of Geophysics and Engineering*, **8**, 21–26.
- Loewer, M., J. Igel, e N. Wagner, 2015, Spectral decomposition of soil electrical and dielectric losses and prediction of in situ gpr performance: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **9**, 212–220.
- Martins, E., 2000, Petrografia, mineralogia e geomorfologia de rególicos lateríticos no distrito federal: Universidade de Brasília–UnB, Brasília.
- Menezes, P. H. B. J., 2010, Avaliação do efeito das ações antrópicas no processo de escoamento superficial e assoreamento na bacia do lago paranoá.
- Menke, W., 1984, Geophysical data analysis: Discrete inverse theory: Academic press.
- Moser, T., 1991, Shortest path calculation of seismic rays: *Geophysics*, **56**, 59–67.
- Natterer, F., e F. Wübbeling, 2001, Mathematical methods in image reconstruction: SIAM.
- Neumann, M. R. B., 2012, Mapeamento digital de solos, no distrito federal.
- Park, S. K., e G. P. Van, 1991, Inversion of pole-pole data for 3-d resistivity structure beneath arrays of electrodes: *Geophysics*, **56**, 951–960.
- Pimentel, M. M., J. B. Rodrigues, M. E. S. DellaGiustina, S. Junges, M. Matteini, e R. Armstrong, 2011, The tectonic evolution of the neoproterozoic brasilia belt, central brazil, based on shrimp and la-icpms u–pb sedimentary provenance data: a review: *Journal of South American Earth Sciences*, **31**, 345–357.
- Pirzadeh, S., 1993, Structured background grids for generation of unstructured grids by advancing-front method: *AIAA journal*, **31**, 257–265.
- Podvin, P., e I. Lecomte, 1991, Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools: *Geophysical Journal International*, **105**, 271–284.
- Ramachandran, P., e G. Varoquaux, 2011, Mayavi: 3d visualization of scientific data: *Computing in Science & Engineering*, **13**, 40–51.
- Rawlinson, N., S. Pozgay, e S. Fishwick, 2010, Seismic tomography: a window into deep earth: *Physics of the Earth and Planetary Interiors*, **178**, 101–135.
- Rawlinson, N., e M. Sambridge, 2003, in Seismic travelttime tomography of the crust and lithosphere, **46**, 81–198.
- Reis Rodrigues, V. H. S., e A. Bassrei, 2015, Aplicação da tomografia de tempos de trânsito a dados do campo de miranga, bacia do recôncavo: 14th International Congress

- of the Brazilian Geophysical Society & EXPOGEF, Rio de Janeiro, Brazil, 3–6 August 2015, Brazilian Geophysical Society, 1185–1190.
- Röhm, A. H., H. Bijwaard, W. Spakman, e J. Trampert, 2000, Effects of arrival time errors on traveltime tomography: *Geophysical Journal International*, **142**, 270–276.
- Ronczka, M., K. Hellman, T. Günther, R. Wisén, e T. Dahlin, 2017, Electric resistivity and seismic refraction tomography: a challenging joint underwater survey at äspö hard rock laboratory: *Solid Earth*, **8**, 671–682.
- Rücker, C., T. Günther, e F. M. Wagner, 2017, pyGIMLi: An open-source library for modelling and inversion in → geophysics: *Computers and Geosciences*, **109**, 106–123.
- Santamarina, J. C., e D. Fratta, 1998, Introduction to discrete signals and inverse problems in civil engineering.
- Scales, J. A., A. Gersztenkorn, e S. Treitel, 1988, Fast ip solution of large, sparse, linear systems: Application to seismic travel time tomography: *Journal of Computational Physics*, **75**, 314–333.
- Strang, G., 1980, Linear algebra and its applications.
- Teng, S.-H., e C. W. Wong, 2000, Unstructured mesh generation: Theory, practice, and perspectives: *International Journal of Computational Geometry & Applications*, **10**, 227–266.
- Wagner, F. M., T. Günther, C. Schmidt-Hattenberger, e H. Maurer, 2015, Constructive optimization of electrode locations for target-focused resistivity monitoring: *Geophysics*, **80**, E29–E40.
- Wang, Y., R. E. White, e R. G. Pratt, 2000, Seismic amplitude inversion for interface geometry: practical approach for application: *Geophysical Journal International*, **142**, 162–172.
- Worthington, M., 1984, An introduction to geophysical tomography: *First Break*, **2**.
- Zhou, B., e S. A. Greenhalgh, 2003, Crosshole seismic inversion with normalized full-waveform amplitude data: *Geophysics*, **68**, 1320–1330.

## A pyGIMLi: Biblioteca *TravelTimeManager* utilizada para o desenvolvimento do *Software*

Listing 1: Código-fonte para pygimli.physics.traveltime.TravelTimeManager

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

"""Class for managing first arrival travel time inversions"""
import os
import numpy as np
import matplotlib.pyplot as plt

from matplotlib.collections import LineCollection

import pygimli as pg
from pygimli.frameworks import MeshMethodManager

from pygimli.utils import getSavePath
from .modelling import TravelTimeDijkstraModelling

class TravelTimeManager(MeshMethodManager):
    """Manager for refraction seismics (traveltime
    tomography).

    TODO Document main members and use default
    MethodManager interface
    e.g., self.inv, self.fop, self.paraDomain, self.mesh,
    self.data
    """

    def __init__(self, data=None, **kwargs):
        """Create an instance of the Traveltime
        manager.

        Parameters
        -----
        data: :gimliapi:`GIMLI::DataContainer` | str
        You can initialize the Manager with data or
        give them a dataset
        when calling the inversion.
        """
        self._useFMM = False
        self.secNodes = 2 # default number of
                         secondary nodes for inversion

        super(TravelTimeManager, self).__init__(data=
```

```

        data, **kwargs)

    self.inv.dataTrans = pg.trans.Trans()

@property
def velocity(self):
    """Return velocity vector (the inversion
    model)."""
    # we can check here if there was an inversion
    # run
    return self.fw.model # shouldn't it be the
    inverse?

def createForwardOperator(self, **kwargs):
    """Create default forward operator for
    Traveltime modelling.

    You want your Manager use a special forward
    operator you can add them
    here on default Dijkstra is used.
    """
    fop = TravelTimeDijkstraModelling(**kwargs)
    return fop

def load(self, fileName):
    """Load any supported data file."""
    self.data = pg.physics.traveltime.load(
        fileName)
    return self.data

def createMesh(self, data=None, **kwargs):
    """Create default inversion mesh.

    Inversion mesh for travelttime inversion does
    not need boundary region.
    """
    d = data or self.data

    if d is None:
        pg.critical('Please provide a data file for
                    mesh generation')

    return pg.meshTools.createParaMesh(d.sensors
        (),
        boundary=0, **kwargs)

def checkData(self, data):
    """Return data from container."""

```

```

        if isinstance(data, pg.DataContainer):
            if not data.haveData('t'):
                pg.critical('DataContainer has no "t" values.')
            )
        return data['t']

    return data

def checkError(self, err, dataVals):
    """Return relative error."""
    if isinstance(err, pg.DataContainer):
        if not err.haveData('err'):
            pg.error('DataContainer has no "err" values.
                      Fallback to 3%')
        return np.ones(err.size()) * 0.03
    return err['err'] / dataVals

    return err

def applyMesh(self, mesh, secNodes=None,
             ignoreRegionManager=False):
    """Apply mesh, i.e. set mesh in the forward
       operator class."""
    if secNodes is None:
        secNodes = self.secNodes

    self.fop._refineSecNodes = secNodes
    if secNodes > 0:
        if ignoreRegionManager:
            mesh = self.fop.createRefinedFwdMesh(mesh)

    self.fop.setMesh(mesh, ignoreRegionManager=
                    ignoreRegionManager)

def simulate(self, mesh, scheme, slowness=None, vel=
            None, seed=None,
            secNodes=2, noiseLevel=0.0, noiseAbs=0.0, **kwargs):
    """Simulate traveltime measurements.

       Perform the forward task for a given mesh, a
       slowness distribution (per
       cell) and return data (travelttime) for a
       measurement scheme.

    Parameters
    -----
    mesh : :gimliapi:`GIMLI::Mesh`
        Mesh to calculate for or use the last known

```

```

    mesh.
scheme: :gimliapi:'GIMLI::DataContainer'
Data measurement scheme needs 's' for shot
    and 'g' for geophone
data token.
slowness : array(mesh.cellCount()) | array(N,
    mesh.cellCount())
Slowness distribution for the given mesh
cells can be:

* a single array of len mesh.cellCount()
* a matrix of N slowness distributions of len
    mesh.cellCount()
* a res map as [[marker0, res0], [marker1,
    res1], ...]
vel : array(mesh.cellCount()) | array(N, mesh
    .cellCount())
Velocity distribution for the given mesh
cells.
Will overwrite given slowness.
secNodes: int [2]
Number of refinement nodes to increase
accuracy of the forward
calculation.
noiseLevel: float [0.0]
Add relative noise to the simulated data.
    noiseLevel*100 in %
noiseAbs: float [0.0]
Add absolute noise to the simulated data in
ms.
seed: int [None]
Seed the random generator for the noise.

```

#### Keyword Arguments

-----

returnArray: [False]

Return only the calculated times.

verbose: [self.verbose]

Overwrite verbose level.

\*\*kwargs

Additional kwargs ...

#### Returns

-----

t : array(N, data.size()) | DataContainer

The resulting simulated travel time values.

Either one column array or matrix in case of
slowness matrix.

```

"""
verbose = kwargs.pop('verbose', self.verbose)

fop = self.fop
fop.data = scheme
fop.verbose = verbose

if mesh is not None:
    self.applyMesh(mesh, secNodes=secNodes,
                   ignoreRegionManager=True)

if vel is not None:
    slowness = 1/vel

if slowness is None:
    pg.critical("Need some slowness or velocity
                 distribution for"
                 " simulation.")

if len(slowness) == self.fop.mesh().cellCount
():
    t = fop.response(slowness)
else:
    print(self.fop.mesh())
    print("slowness: ", slowness)
    pg.critical("Simulate called with wrong
                 slowness array.")

ret = pg.DataContainer(scheme)
ret.set('t', t)

if noiseLevel > 0 or noiseAbs > 0:
    if not ret.allNonZero('err'):
        ret.set('t', t)
        err = noiseAbs + t * noiseLevel
        ret.set('err', err)

    pg.verbose("Absolute error estimates (min:max
               ) {0}:{1}".format(
               min(ret('err')), max(ret('err'))))

    t += pg.randn(ret.size(), seed=seed) * ret(
        'err')
    ret.set('t', t)

if kwargs.pop('returnArray', False) is True:
    return t

```

```

        return ret

    def invert(self, data=None, useGradient=True, vTop
              =500, vBottom=5000,
              secNodes=2, **kwargs):
        """Invert data.

        Parameters
        -----
        data : pg.DataContainer()
            Data container with at least SensorIndieces ,
            's' and
            data values 't' (traveltime in ms) and 'err'
            (absolute error in ms)
        useGradient: bool [True]
            Use a gradient like starting model suited for
            standard flat
            earth cases. [Default]
            For cross tomography geometry you should set
            this to False for a
            non gradient startind model.
        vTop: float
            Top velocity for gradient stating model.
        vBottom: float
            Bottom velocity for gradient stating model.
        secNodes: int [2]
            Amount of secondary nodes used for ensure
            accuracy of the forward
            operator.

        Keyword Arguments
        -----
        ** kwargs:
            Inversion related arguments:
            See :py:mod:`pygimli.frameworks.
                MeshMethodManager.invert`
        """
        mesh = kwargs.pop('mesh', None)

        self.secNodes = secNodes

        if 'limits' in kwargs:
            if kwargs['limits'][0] > 1:
                tmp = kwargs['limits'][0]
                kwargs['limits'][0] = 1.0 / kwargs['limits',
                                                ][1]
                kwargs['limits'][1] = 1.0 / tmp
                pg.verbose('Switching velocity limits to'

```

```

        slowness limits.,
        kwargs['limits'])

    if useGradient:
        self.fop._useGradient = [vTop, vBottom]
    else:
        self.fop._useGradient = None

    slowness = super().invert(data, mesh, **
                               kwargs)
    velocity = 1.0 / slowness
    self.fw.model = velocity
    return velocity

def drawRayPaths(self, ax, model=None, **kwargs):
    """Draw the ray paths for model or last
    model.

    If model is not specified, the last
    calculated Jacobian is used.

    Parameters
    -----
    model : array
        Velocity model for which to calculate and
        visualize ray paths (the
        default is model for last Jacobian
        calculation in self.velocity).
    ax : matplotlib.axes object
        To draw the model and the path into.
    **kwargs : type
        Additional arguments passed to LineCollection
        (alpha, linewidths,
        color, linestyles).

    Returns
    -----
    lc : matplotlib.LineCollection
    """
    if model is not None:
        if self.fop.jacobian().size() == 0 or model
           != self.model:
            self.fop.createJacobian(1/model)
        else:
            model = self.model

        _ = kwargs.setdefault("color", "w")
        _ = kwargs.setdefault("alpha", 0.5)

```

```

        _ = kwargs.setdefault("linewidths", 0.8)

shots = self.fop.data.id("s")
recei = self.fop.data.id("g")

segss = []
for s, g in zip(shots, recei):
    wi = self.fop.way(s, g)
    points = self.fop._core.mesh().positions(
        withSecNodes=True)[wi]
    segss.append(np.column_stack((pg.x(points), pg
        .y(points)))))

lc = LineCollection(segss, **kwargs)
ax.add_collection(lc)

return lc

def showRayPaths(self, model=None, ax=None,
**kwargs):
    """Show the model with ray paths for given
    model.

    If not model specified, the last calculated
    Jacobian is taken.

    Parameters
    -----
    model : array
        Velocity model for which to calculate and
        visualize ray paths (the
        default is model for last Jacobian
        calculation in self.velocity).
    ax : matplotlib.axes object
        To draw the model and the path into.
    **kwargs : type
        forward to drawRayPaths

    Returns
    -----
    ax : matplotlib.axes object
    cb : matplotlib.colorbar object (only if
        model is provided)

    Examples
    -----
    >>> # No reason to import matplotlib
    >>> import pygimli as pg

```

```

>>> from pygimli.physics import
    TravelTimeManager
>>> from pygimli.physics.traveltime import
    createRAData
>>>
>>> x, y = 8, 6
>>> mesh = pg.createGrid(x, y)
>>> data = createRAData([(0,0)] + [(x, i) for
    i in range(y)],
    ...                                         shotDistance=y+1)
>>> data.set("t", pg.Vector(data.size(), 1.0)
    )
>>> tt = TravelTimeManager()
>>> tt.fop.setData(data)
>>> tt.applyMesh(mesh, secNodes=10)
>>> ax, cb = tt.showRayPaths(showMesh=True,
    diam=0.1)
"""
if model is None:
    if self.fop.jacobian().size() == 0:
        self.fop.mesh() # initialize any meshes ...
        just to be sure is 1
    model = pg.Vector(self.fop.regionManager().
        parameterCount(),
        1.0)
else:
    model = self.model

ax, cbar = self.showModel(ax=ax, model=model,
    showMesh=kwargs.pop('showMesh', None),
    diam=kwargs.pop('diam', None))

self.drawRayPaths(ax, model=model, **kwargs)

return ax, cbar

def rayCoverage(self):
    """Ray coverage, i.e. summed raypath lengths.
    """
    return self.fop.jacobian().transMult(
        np.ones(self.fop.jacobian().rows()))

def standardizedCoverage(self):
    """Standardized coverage vector (0|1) using
    neighbor info."""
    coverage = self.rayCoverage()
    C = self.fop.constraintsRef()
    return np.sign(np.absolute(C.transMult(C *

```

```

        coverage)))

def showCoverage(self, ax=None, name='coverage', **kwargs):
    """Show the ray coverage in log-scale."""
    if ax is None:
        fig, ax = plt.subplots()

    cov = self.rayCoverage()
    return pg.show(self.fop.paraDomain,
                  pg.log10(cov+min(cov[cov > 0])*5), ax=ax,
                  coverage=self.standardizedCoverage(), **kwargs)

def saveResult(self, folder=None, size=(16, 10),
               verbose=False, **kwargs):
    """Save the results in a specified (or date-
       time derived) folder.

    Saved items are:
    * Resulting inversion model
    * Velocity vector
    * Coverage vector
    * Standardized coverage vector
    * Mesh (bms and vtk with results)

    Args
    ----
    path: str[None]
        Path to save into. If not set the name is
        automatically created
    size: (float, float) (16,10)
        Figure size.

    Keyword Args
    -----
    Will be forwarded to showResults

    Returns
    -----
    str:
        Name of the result path.
    """
    subfolder = self.__class__.__name__
    path = getSavePath(folder, subfolder)

    if verbose:
        pg.info('Saving refraction data to: {}'.format(path))

```

```

        format(path))

np.savetxt(os.path.join(path, 'velocity.
    vector'),
self.velocity)
np.savetxt(os.path.join(path, 'velocity-cov.
    vector'),
self.rayCoverage())
np.savetxt(os.path.join(path, 'velocity-scov.
    vector'),
self.standardizedCoverage())

m = pg.Mesh(self.paraDomain)

m['Velocity'] = self.paraModel(self.velocity)
m['Coverage'] = self.rayCoverage()
m['S_Coverage'] = self.standardizedCoverage()
m.exportVTK(os.path.join(path, 'velocity'))
m.saveBinaryV2(os.path.join(path, 'velocity-
    pd'))
self.fop.mesh().save(os.path.join(path, 'velocity-
    mesh'))

np.savetxt(os.path.join(path, 'chi.txt'),
self.inv.chi2History)

fig, ax = plt.subplots()
self.showResult(ax=ax, cov=self.
    standardizedCoverage(), **kwargs)
fig.set_size_inches(size)
fig.savefig(os.path.join(path, 'velocity.pdf',
    ), bbox_inches='tight')
pg.plt.close(fig)
return path

```