

## 5

## Segmentação do grafo de falhas

### 5.1.

#### Introdução

Como visto no capítulo anterior, a utilização do algoritmo Growing Neural Gas sobre um atributo de falha permite a geração de um grafo que acompanha a estrutura de probabilidades de ocorrência de falhas. O grafo assim gerado é fortemente esparso, apresentando uma alta densidade de nós apenas no entorno das regiões de valores altos do indicador de falha. A questão de como individualizar cada uma das superfícies presentes nos dados (problema de Extração de Regiões de Falha enunciado na seção 3.5) é o tema deste capítulo.

Uma estratégia para abordar o problema seria associar pesos às arestas e aplicar um algoritmo de segmentação de grafos. Essa é a estratégia empregada por Hale e Emanuel, conforme visto na seção 3.4. Entretanto, uma falha não corresponde a um evento sísmico coerente, mas a uma descontinuidade em uma zona coerente. Os atributos de falha tentam realçar tais descontinuidades (ver seção 2.2). Porém, o fato de se trabalhar com descontinuidades dos dados leva a dois tipos de problemas nos indicadores de falha resultantes. Primeiro, que as superfícies de falha são vistas mais como “tendências” a falhas do que como superfícies contínuas bem definidas (Pedersen et al., 2002). Dito de outra forma, as falhas vistas através desses atributos resultam, geralmente, em “rasgos” descontínuos de espessura e intensidade variável. Na estratégia de Hale, antes da geração do grafo, o atributo de falhas é submetido a uma filtragem direcional de forma a condicionar os dados, isto é, eliminar a influência de ruídos e reforçar a continuidade do atributo. Na verdade, a filtragem utilizada por Hale faz muito mais do que isso: as estruturas presentes nos dados são efetivamente afinadas por um processo semelhante ao algoritmo de Canny. Assim, o problema de descontinuidade e variabilidade de espessura é resolvido por um método de processamento de imagens.

Entretanto, além desse, um segundo problema deve ser levado em consideração: vários outros tipos de evento podem ter uma assinatura semelhante a uma falha, por exemplo bordo de canal, variações locais do sinal

sísmico ao longo de horizontes, chaminé de gás ou zona ruidosa (Labrunye, 2004). Todos estes tipos de eventos geram descontinuidades em uma zona coerente dos dados sísmicos. Ao observar o cubo de dados sísmicos, o geólogo pode, apesar de tudo, diferenciar uma falha dessas outras estruturas pela análise de sua geometria: em geral, uma falha é uma superfície mais ou menos vertical de curvatura pequena. Esse tipo de problema não é considerado na abordagem de Hale-Emanuel.

Nesta tese, pretende-se atacar os dois problemas no contexto do grafo de falhas. Uma estratégia de segmentação em duas etapas é proposta de forma a gerar um conjunto de subgrafos, cada um associado a uma das superfícies de falha presente nos dados. A primeira etapa se baseia nos tamanhos das arestas, desconectando regiões ligadas por arestas de tamanho incompatível com uma ou ambas as regiões. Na linguagem do capítulo anterior, essa etapa elimina arestas pela identificação de regiões cuja distribuição de vértices não é densa. Tal etapa é chamada neste trabalho de eliminação de arestas **frágeis**. A qualificação de frágil para esse tipo de aresta deriva do fato de ser uma aresta maior do que outras da sua vizinhança, atravessando uma região com distribuição de probabilidade também menor do que nas regiões vizinhas, possuindo, portanto, uma baixa densidade linear de probabilidade.

A segunda etapa se baseia na estimação da orientação local dos nós do grafo, mantendo conectados nós em que a orientação local é compatível com uma superfície pseudo-vertical de baixa curvatura.

Por fim, mesmo com esse duplo processo de segmentação do grafo, não se espera obter como resultado um grafo totalmente afinado que corresponda a uma superfície, isto é, que tenha uma espessura zero. A questão relativa à geração de uma superfície para modelagem e visualização através de uma malha de polígonos será tratada no Capítulo 6.

## 5.2. Segmentação de grafos

Algumas soluções simples podem ser pensadas para a eliminação de arestas frágeis. Por exemplo, pode-se calcular a média  $\mu$  e desvio padrão  $\sigma$  dos tamanhos de todas as arestas e desconectar arestas com tamanhos maiores do que um valor limite dependente de  $\mu$  e  $\sigma$ . Tal abordagem, entretanto, ignora a possibilidade de se ter regiões com diferentes variabilidades de tamanho de arestas. Uma solução mais adaptativa consistiria em comparar o tamanho da

aresta com os tamanhos de arestas nas vizinhanças dos dois nós conectados. Entretanto, em situações como a da Figura 5-1 abaixo, essa estratégia pode falhar. Observando a Figura, é intuitivo dizer que a aresta vermelha que liga os nós *a* e *b* é uma aresta frágil. Entretanto, a vizinhança imediata do nó *a* contém também outra aresta frágil, além da aresta vermelha, o que torna maior a média dos tamanhos em volta do nó *a*. Assim, é necessária a busca de métodos mais robustos de segmentação de grafos para tratar o problema.

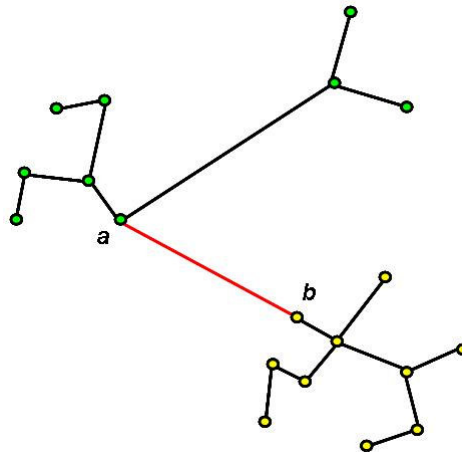


Figura 5-1: Grafo exibindo uma aresta frágil (em vermelho). A vizinhança do nó *a* contém outra aresta frágil, o que eleva a média local dos tamanhos de aresta.

Grafos são utilizados para modelar diferentes problemas. Como consequência, a ideia de particionar ou segmentar grafos surge de diferentes maneiras conforme o contexto em questão. Por exemplo, em computação paralela, são utilizados grafos para modelar um problema computacional. Uma partição do grafo em subgrafos leva a uma decomposição dos dados e/ou tarefas e os subgrafos podem ser mapeados em processadores de um multiprocessador (Pothén, 1997). O particionamento pode ser feito com diferentes objetivos: o objetivo de gerar um baixo custo de comunicação se reflete em minimizar o número de arestas de corte; o balanceamento de carga, por sua vez, implica em prescrever o número de vértices de cada subgrafo dentro de uma tolerância. Outros objetivos são possíveis.

Um contexto mais próximo do problema de eliminação de arestas frágeis é o de segmentação de imagens. A segmentação de uma imagem é a sua divisão em regiões (componentes, fragmentos ou segmentos), de forma que cada uma das regiões seja homogênea em algum sentido, ao mesmo tempo em que nenhuma união de quaisquer duas regiões adjacentes será homogênea. Por

homogêneo, entende-se que nenhum segmento exibe mudanças abruptas no seu interior.

Grafos são também utilizados para modelar o problema de segmentação de imagens. Nesse caso, cada nó  $v_i \in V$  do grafo corresponde a um pixel da imagem e as arestas em  $A$  conectam certos pares de pixels vizinhos. Um peso  $w_i$  é associado a cada aresta, baseado em alguma propriedade dos pixels que a aresta conecta, como, por exemplo, a intensidade da imagem. O peso da aresta é uma medida (normalmente positiva) da dissimilaridade entre os nós. Assim, espera-se que nós de um mesmo segmento tenham arestas internas com pesos relativamente baixos (isto é, que os nós sejam semelhantes), enquanto arestas que ligam nós de segmentos diferentes tenham pesos com valores maiores (ou seja, ligam nós diferentes).

O conceito de segmentação de imagens pode ser definido mais formalmente da seguinte maneira: se  $I$  é o conjunto de todos os pixels e  $D$  é um predicado lógico que identifica se uma região de  $I$  é homogênea, a segmentação é uma partição do conjunto  $I$  em um conjunto de subconjuntos ou regiões  $S_1, S_2, \dots, S_n$  tal que:

$$\begin{aligned} \bigcup_{i=1}^n S_i &= I \\ S_i \cap S_j &= \emptyset, \quad \forall i \\ D(S_i) &= \text{VERDADE}, \quad \forall i \\ D(S_i \cup S_j) &= \text{FALSO}, \quad \forall S_i, S_j \text{ adjacente}, i \neq j. \end{aligned}$$

A definição não estabelece que a partição ou o número de segmentos seja único. O conceito é exatamente o mesmo para o caso de um dado volumétrico escalar, bastando substituir pixel por voxel.

O conceito também pode ser aplicado a grafos não-dirigidos em geral. Seja  $G=(V,A)$  um grafo não-dirigido, onde  $V$  é o conjunto de vértices e  $A$  o de arestas. Uma segmentação de  $G$  é uma partição de  $V$  em segmentos  $S_i$ , onde cada segmento corresponde a uma componente conexa de um subgrafo  $G'=(V,A')$  de  $G$  ( $A' \subseteq A$ ). Em outras palavras, qualquer segmentação é induzida por um subconjunto de arestas em  $A$ . No caso da eliminação de arestas frágeis, esse subconjunto é o conjunto de todas as arestas menos as frágeis. Pode-se, então, definir o conceito de aresta frágil a partir de uma segmentação previamente obtida: uma aresta é considerada frágil se e somente se conectar dois segmentos diferentes.

Existe uma vasta literatura sobre segmentação de imagens modeladas por grafos, cujos primeiros trabalhos remontam a mais de 30 anos atrás. Em particular, um dos primeiros métodos baseado em grafos foi proposto por Zahn em 1971 (Zahn, 1971). Esse método é baseado na Árvore Geradora Mínima (AGM) do grafo<sup>38</sup> e foi aplicado para segmentar nuvem de pontos e para segmentar imagem. Para a segmentação de imagem os pesos das arestas do grafo são baseados nas diferenças entre as intensidades dos pixels, enquanto para nuvem de pontos, os pesos das arestas são baseados nas distâncias entre os pontos.

O método de segmentação de Zahn parte da AGM do grafo e utiliza como critério de segmentação a eliminação das arestas da árvore que tenham pesos altos. Zahn chamou tais arestas de arestas **inconsistentes**<sup>39</sup>. Estas são identificadas através da comparação do tamanho da aresta com o tamanho médio local em torno de cada um dos nós por ela ligados. Zahn utiliza valores de limiar fixos nessa comparação. Isto impede de tratar adequadamente situações onde convivem, num mesmo grafo, regiões de baixa variabilidade e regiões de alta variabilidade. Além disso, situações como a da Figura 5-1 podem ser tratadas de forma inadequada.

Uma estratégia utilizada em alguns algoritmos consiste em identificar o corte mínimo no grafo. Nesses casos, os pesos do grafo apresentam informações sobre a similaridade entre os nós. A identificação de um corte apenas divide o grafo em duas partes. As possíveis segmentações são obtidas pela aplicação sucessiva do método para cada uma das partes obtidas. Na verdade, a definição de corte deve ser alterada de forma a evitar a tendência de geração de componentes muito pequenas. Tal tipo de problema foi abordado pelo critério de corte normalizado proposto por Shi e Malik (Shi e Malik, 2000). Como em outros métodos, a solução desse algoritmo passa pela identificação dos autovetores de uma matriz. A classe de métodos que se baseiam na identificação de autovetores de uma matriz é dita de métodos espectrais. Estes se baseiam na divisão em blocos diagonais de uma matriz derivada da matriz de adjacência do grafo ponderado, sendo os segmentos então associados aos blocos (Weiss, 1999). Como um método de identificação de corte mínimo, ele

---

<sup>38</sup> A AGM de um grafo  $G(V,A)$  é o subgrafo  $G'(V,A')$  onde  $A'$  é subconjunto de  $A$  e acíclico e cuja soma dos pesos é mínima.

<sup>39</sup> Observe que a expressão aresta inconsistente proposta por Zahn corresponde ao que foi chamado anteriormente, neste trabalho, de aresta frágil.

provê apenas uma caracterização de cada corte, gerando uma hierarquia de segmentações e não uma segmentação final.

À semelhança do corte normalizado, o método de aglomeração estocástica (*stochastic clustering*) proposto por Gdalyahu e Weinshall (Gydalyahu et al., 2001) é também um método hierárquico, mas, nesse caso, a hierarquia é construída de baixo para cima, isto é, inicialmente cada pixel é visto como um segmento separado e a aplicação do algoritmo gera aglomerações de pixels.

O método proposto por Felzenszwalb e Huttenlocher (2004) é também um método baseado na Árvore Geradora Mínima. Mas, diferente de métodos anteriores, como o de Zahn, que atua sobre a árvore já calculada, esse método atua sobre o próprio processo de construção da árvore. O método estabelece um critério adaptativo que depende de propriedades de cada segmento em construção como um todo, isto é, depende de propriedades não estritamente locais. Assim, o algoritmo é capaz de capturar aspectos globais da imagem, o que outros algoritmos baseados em AGM não conseguem. Por outro lado, mantém o excelente desempenho computacional, sendo executado em tempo  $O(|A| \log |A|)$ , onde  $|A|$  é o número de arestas do grafo.

A escolha do método de Felzenszwalb e Huttenlocher (daqui em diante chamado de algoritmo FH) para a eliminação das arestas frágeis se deve à sua simplicidade de implementação e bom desempenho computacional. Além disso, os resultados obtidos se mostraram dentro do esperado nos testes executados.

### 5.3.

#### **Primeiro passo: segmentação baseada nos tamanhos das arestas**

##### 5.3.1.

##### **Algoritmo FH**

O algoritmo FH (Felzenszwalb et al., 2004) define explicitamente o predicado  $D$  que avalia se existe ou não evidência de uma fronteira entre duas componentes em uma segmentação. O predicado é baseado em uma medida de dissimilaridade entre nós vizinhos pertencentes a componentes diferentes. Tal medida é avaliada em relação ao grau de dissimilaridade presente internamente em cada uma das duas componentes, sendo, portanto, adaptativa às características dos dados.

Define-se **diferença interna** de um segmento  $S \subseteq V$  como sendo o maior peso de uma aresta da Árvore Geradora Mínima do segmento,  $AGM(S, A)$ ; isto é:

$$\text{Int}(S) = \max_{a \in \text{AGM}(S,A)} w(a),$$

onde  $w(a)$  é o peso da aresta  $a$ . A idéia por trás dessa definição é que o segmento  $S$  se mantém conectado quando arestas de peso no máximo  $\text{Int}(S,A)$  são consideradas.

Define-se **diferença-entre** dois segmentos  $S_1, S_2 \subseteq V$  como o menor peso das arestas que ligam os dois segmentos; isto é:

$$\text{Dif}(S_1, S_2) = \min_{v_i \in S_1, v_j \in S_2, (v_i, v_j) \in A} w(v_i, v_j)$$

No caso de não haver aresta ligando os dois segmentos, define-se  $\text{Dif}(S_1, S_2) = \infty$ .

Obviamente, espera-se que o predicado  $D$  reconheça a evidência de uma fronteira entre  $S_1$  e  $S_2$  quando  $\text{Int}(S_1) < \text{Dif}(S_1, S_2)$  e  $\text{Int}(S_2) < \text{Dif}(S_1, S_2)$ . Entretanto, no caso em que se está em um processo de construção dos segmentos, é preciso definir uma margem de tolerância, de forma a permitir a agregação de segmentos cujas diferenças internas tenham valores próximos da diferença-entre. Assim, o predicado é definido como:

$$D(S_1, S_2) = \begin{cases} \text{VERDADE, se } \text{Dif}(S_1, S_2) > \text{MInt}(S_1, S_2) \\ \text{FALSO, caso contrário} \end{cases}$$

onde  $\text{MInt}$  é uma avaliação da diferença interna com **tolerância**  $\tau$ :

$$\text{MInt}(S_1, S_2) = \min(\text{Int}(S_1) + \tau(S_1), \text{Int}(S_2) + \tau(S_2)).$$

Observe que, para segmentos pequenos,  $\text{Int}(S)$  não é uma boa estimativa das características locais dos dados; no caso extremo em que só se tem um único nó no segmento ( $|S|=1$ ),  $\text{Int}(S)=0$ . Felzenszwalb e Huttenlocher propõem uma tolerância  $\tau$  que varia com o inverso do número de elementos do segmento em construção:

$$\tau(S) = \frac{k}{|S|}$$

onde  $|S|$  é o número de nós do segmento  $S$  e  $k$  é um parâmetro constante. Assim, para segmentos pequenos se exige uma grande evidência para a formação de uma fronteira. Na prática  $k$  estabelece uma escala de observação, no sentido de que grandes valores de  $k$  causam a geração de segmentos maiores. Por outro lado, é importante observar que  $k$  não é um tamanho mínimo de segmento; tamanhos menores podem ocorrer contanto que existam arestas com pesos suficientemente grandes.

O método FH de segmentação é baseado no processo de construção da AGM; assim, o algoritmo constitui-se de uma pequena variação do algoritmo de

Kruskal (Cormen, 2001). No início do algoritmo de Kruskal, cada nó constitui um conjunto (segmento) separado. O conjunto de arestas é organizado em ordem crescente dos pesos. A cada passo, uma aresta é selecionada e caso seus nós pertençam a conjuntos diferentes, os conjuntos se fundem e a aresta é identificada como uma aresta da AGM. A alteração consiste apenas em só permitir a fusão de dois segmentos quando a aresta selecionada pelo algoritmo violar o predicado D de fronteira<sup>40</sup>. A seguir é apresentado o algoritmo FH:

### Algoritmo de segmentação FH

A entrada é um grafo  $G = (V, A)$ , com  $n$  vértices e  $m$  arestas. A saída é uma segmentação de  $V$  em componentes  $S = (S_1, \dots, S_l)$ .

1. Ordenar  $A$  em  $\pi = (o_1, \dots, o_m)$  em ordem decrescente de pesos.
2. Iniciar com uma segmentação  $S^0$ , onde cada vértice  $v_i$  é o único elemento do seu segmento.
3. Repetir passo 4 para  $q = 1, \dots, m$ .
4. Construir  $S^q$  a partir de  $S^{q-1}$  como a seguir. Sejam  $v_i$  e  $v_j$  os vértices da aresta  $o_q = (v_i, v_j)$ . Seja  $S^{q-1}_i$  o segmento que contém  $v_i$  e  $S^{q-1}_j$  o que contém  $v_j$ . Se  $S^{q-1}_i \neq S^{q-1}_j$  e  $w(o_q) \leq \text{MInt}(S^{q-1}_i, S^{q-1}_j)$  então unir os dois conjuntos, formando  $S^q$ . Caso contrário,  $S^q = S^{q-1}$ .
5. Retornar  $S = S^m$ .

### 5.3.2. Identificação de arestas frágeis via FH

Para aplicar o algoritmo FH ao problema de eliminação de arestas frágeis é necessário associar pesos às arestas do grafo de falhas. Como comentado anteriormente, a idéia de fragilidade de uma aresta está associada ao seu tamanho (na verdade, não em termos absolutos, mas em relação ao conjunto de arestas dos seus vizinhos). Por outro lado, o algoritmo FH, como apresentado acima, assume que os pesos das arestas do grafo estão associados a uma medida de dissimilaridade dos nós ligados. Assim, para utilizar o algoritmo FH no problema de eliminação de arestas frágeis, basta assumir que a medida de dissimilaridade entre dois nós é a distância euclidiana entre eles.

---

<sup>40</sup> O algoritmo FH mantém a estratégia gulosa do algoritmo Kruskal.



Por outro lado, o resultado do algoritmo FH é um conjunto de segmentos  $\{S_1, \dots, S_r\}$ , onde cada segmento possui seu conjunto de nós. As arestas frágeis foram definidas a partir de uma segmentação como arestas que conectam dois segmentos diferentes. Podem, então, ser identificadas e eliminadas percorrendo-se o grafo original com os nós já classificados, após a execução do FH<sup>41</sup>.

Outra forma de obter as arestas frágeis seria pela alteração do passo 4 do algoritmo FH. No caso em que a aresta  $o_q$  é caracterizada como incapaz de conectar segmentos diferentes (isto é,  $w(o_q) > \text{MInt}(S^{q-1}_i, S^{q-1}_j)$ ) ela pode ser identificada como frágil e removida do conjunto de arestas. Além disso, o resultado é o grafo com o conjunto de arestas alterado. Os passos 4 e 5 alterados ficariam:

- 4'. Construir  $S^q$  a partir de  $S^{q-1}$  como a seguir. Sejam  $v_i$  e  $v_j$  os vértices da aresta  $o_q = (v_i, v_j)$ . Seja  $S^{q-1}_i$  o segmento que contém  $v_i$  e  $S^{q-1}_j$  o que contém  $v_j$ . Se  $S^{q-1}_i \neq S^{q-1}_j$  e  $w(o_q) \leq \text{MInt}(S^{q-1}_i, S^{q-1}_j)$  então unir os dois conjuntos, formando  $S^q$ . Caso contrário,  $S^q = S^{q-1}$  e remover a aresta  $o_q$  do conjunto de arestas  $A$ .
- 5'. Retornar  $G = (V, A)$ .

### 5.3.3. Resultados

As Figuras abaixo apresentam alguns resultados da aplicação do algoritmo FH sobre os dados sísmicos de teste, com diferentes valores do parâmetro  $k$ . A Figura 5-2 apresenta o mesmo grafo gerado por GNG apresentado ao final do Capítulo 4, onde se percebe a presença de várias arestas frágeis. A seguir são apresentados os resultados do algoritmo FH para  $k=5$  (Figura 5-3),  $k=15$  (Figura 5-4) e  $k=50$  (Figura 5-5). Nós e arestas de um mesmo segmento são apresentados com mesma cor. A paleta utilizada contém apenas 20 cores e, eventualmente, uma mesma cor pode estar repetida em diferentes segmentos. Nós que ficaram completamente desconectados não são exibidos nas Figuras.

---

<sup>41</sup> Observe que após a eliminação das arestas frágeis, o grafo resultante dificilmente forma uma árvore, apesar do algoritmo FH ser baseado em Kruskal.

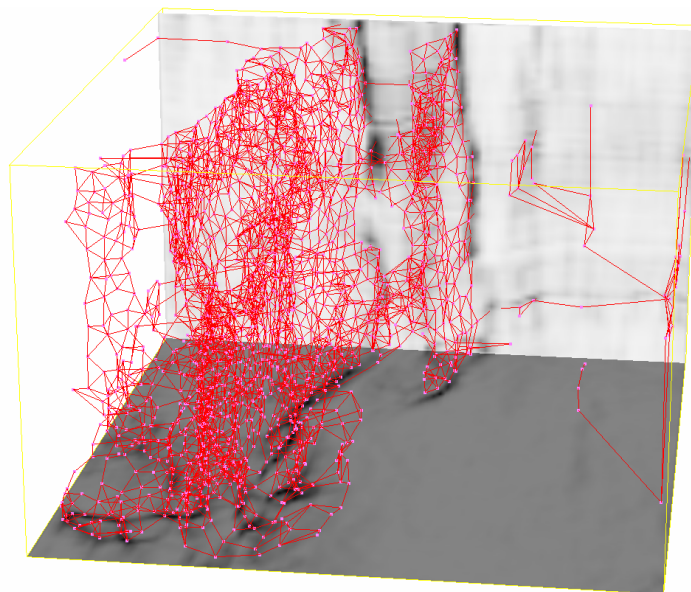


Figura 5-2: Grafo gerado por GNG, com arestas frágeis

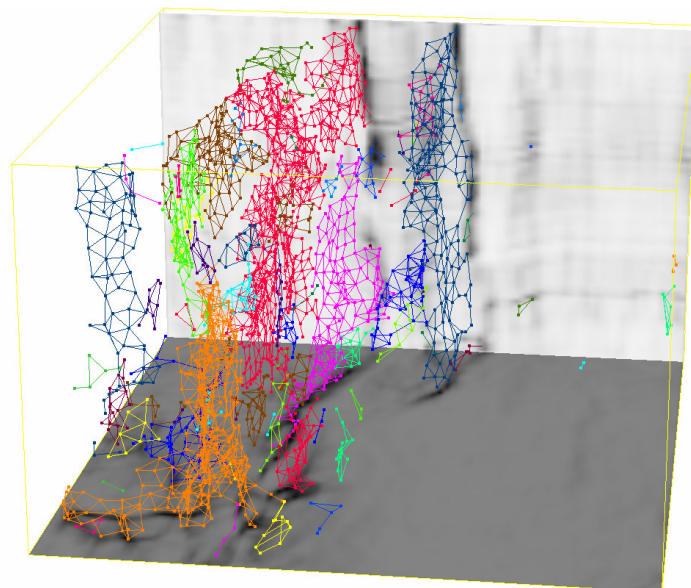


Figura 5-3: Grafo da Figura 5-2 segmentado por FH com  $k=5$ . Nós isolados não são apresentados.

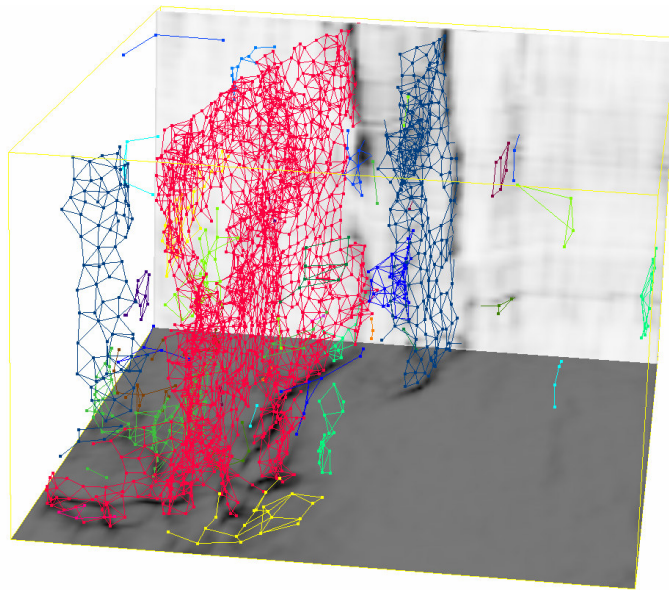


Figura 5-4: Grafo da Figura 5-2 segmentado por FH com  $k=15$ . Nós isolados não são apresentados.

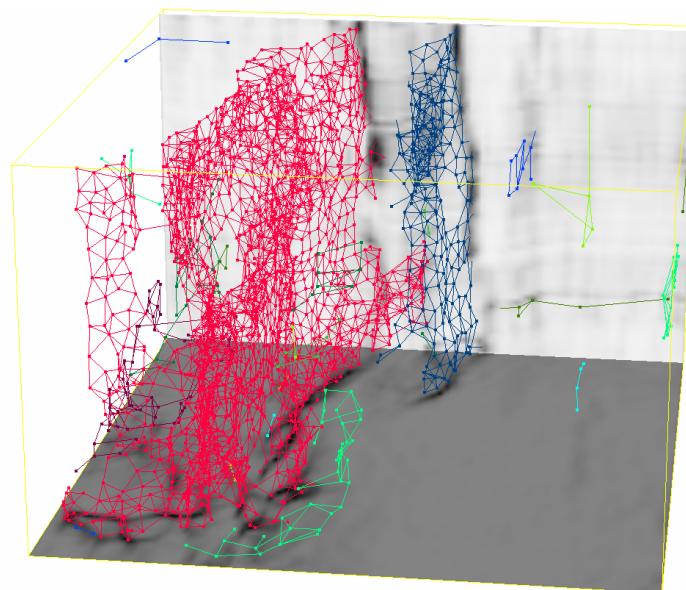


Figura 5-5: Grafo da Figura 5-2 segmentado por FH com  $k=50$ . Nós isolados não são apresentados.

A comparação entre as imagens da Figura 5-3 até a Figura 5-5 mostra uma boa eliminação de arestas frágeis para  $k$  em torno de 15. Como era esperado, um valor pequeno de  $k$ , no caso 5, gera uma grande quantidade de pequenos segmentos, muitos dos quais ainda poderiam ser combinados (*over-segmentation*). Por outro lado, um valor grande de  $k$ , no caso 50, gera poucos segmentos com grande quantidade de nós, tendo anexado no mesmo segmento regiões separadas por arestas frágeis (*under-segmentation*).

Além disso, observando os detalhes exibidos na Figura 5-6, percebe-se que parte da estrutura maior (em vermelho) não tem orientação vertical bem definida. O mesmo ocorre com a estrutura em amarelo. Esse tipo de situação é tratado pelo segundo passo de segmentação.

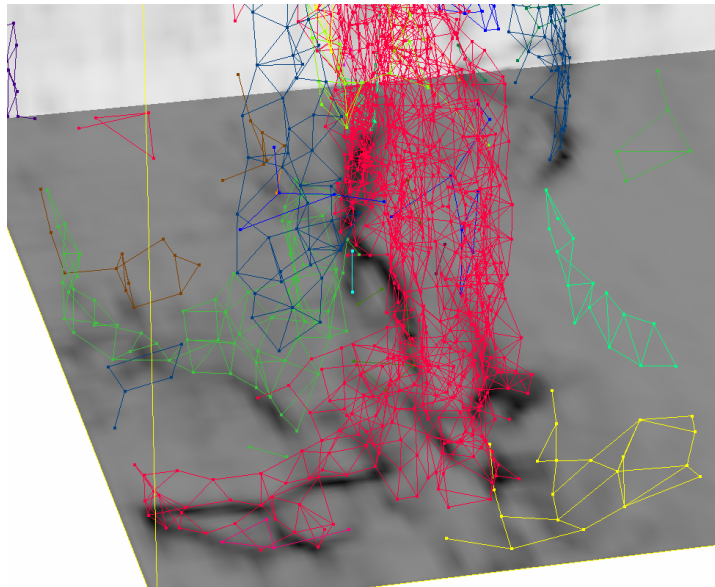


Figura 5-6: Detalhe da Figura 5-5, mostrando a ocorrência de estruturas não verticais.

#### **5.3.4. Eliminação de pontas**

O grafo resultante do passo de eliminação de arestas frágeis pode conter arestas dispostas em linha, formando pontas. De fato, isso ocorre em todas as Figuras de grafos segmentados exibidas nesta seção (ver por exemplo, a Figura 5-6). Tais estruturas não são compatíveis com falhas sísmicas; além disso, são fontes de problemas para uma avaliação da orientação local feita no segundo passo de segmentação. Assim, ao final do primeiro passo de segmentação, deve-se eliminar as arestas do grafo que estão dispostas em linha. Isso pode ser feito de fora para dentro, isto é, varrendo o conjunto de nós do grafo, procurando por nós com uma única aresta. A cada nó encontrado, a aresta e o nó são eliminados e se testa o outro nó conectado pela aresta eliminada. Caso esse nó agora tenha também uma única aresta, repete-se o procedimento de eliminação.

#### 5.4.

#### **Segundo passo: segmentação de grafos baseada na orientação local**

O problema da segmentação do grafo baseada na orientação local da estrutura remete ao problema de extração de superfícies suaves em nuvens de pontos geradas por scanners a laser ou outro tipo de equipamento que gere dados de profundidade ou distância. Esse tipo de dado também é conhecido como *range image*; o nome deriva do fato de que a nuvem de pontos usada nas primeiras pesquisas com scanners a laser era capturada em uma grade regular e armazenada em imagens *raster* (Vosselman et al., 2004). Aplicações como a extração da superfície nua da Terra a partir de dados gerados por equipamento a laser aerotransportado, engenharia reversa de sítios industriais e a produção de modelos tridimensionais de cidades dependem do sucesso de um passo de segmentação automática dos dados (Vosselman et al., 2004). Existem três classes principais de algoritmos de segmentação para *range images*: baseada em bordas, baseada em varredura de linhas e baseada em superfícies (Rabanni et al., 2006). Algoritmos baseados em borda são compostos por dois estágios: detecção das bordas que contornam as diferentes regiões, seguida pelo agrupamento dos pontos no interior de cada borda. Algoritmos baseados em varredura de linha se baseiam no fato de que uma varredura de linha (*scanline*) de qualquer plano no espaço tridimensional irá resultar em uma linha no espaço. No primeiro estágio, os segmentos de linha são detectados; em seguida, linhas adjacentes com propriedades similares são agrupadas. A classe dos algoritmos de segmentação baseados em superfície é a que mais se aproxima da questão da segmentação do grafo de falhas. Nesse tipo de segmentação, superfícies suaves são extraídas pelo agrupamento de pontos próximos que compartilham alguma propriedade local, como a direção de uma estimativa do vetor normal.

Neste trabalho, é proposta uma segmentação baseada na avaliação da orientação local de cada nó do grafo. Diferente da situação de nuvem de pontos, no caso em questão, a informação de vizinhança já se encontra disponível (nas arestas do grafo). O processo de cálculo da normal também produz uma estimativa da confiabilidade da orientação obtida. De posse dessas duas informações (vetor normal e confiança) para cada nó do grafo, este trabalho propõe um algoritmo de segmentação baseado em crescimento de regiões.

### 5.4.1.

#### Cálculo da orientação local dos nós

A análise dos autovetores e autovalores da matriz de covariância das posições dos nós do grafo em uma vizinhança local permite extrair estimativas dos vetores normais.

Seja  $\mathbf{p}_i$  o vetor posição no  $\mathfrak{R}^3$  de cada vértice  $v_i$  do grafo. Seja  $N_v^k$  a vizinhança do vértice  $v$  de profundidade  $k$ , isto é o conjunto de nós que estão conectados ao nó  $v$  por um caminho com até  $k$  arestas<sup>42</sup>. Seja  $\bar{\mathbf{p}}$  o centróide de uma vizinhança de um vértice  $v$  (de posição  $\mathbf{p}$ ), isto é,

$$\bar{\mathbf{p}} = \frac{1}{|N_v^k|} \sum_{i \in N_v^k} \mathbf{p}_i.$$

A matriz de covariância  $\mathbf{C}$  para um ponto  $\mathbf{p}$  é dada por:

$$\mathbf{C} = \begin{bmatrix} \mathbf{p}_{i_1} - \bar{\mathbf{p}} \\ \dots \\ \mathbf{p}_{i_n} - \bar{\mathbf{p}} \end{bmatrix}^T \begin{bmatrix} \mathbf{p}_{i_1} - \bar{\mathbf{p}} \\ \dots \\ \mathbf{p}_{i_n} - \bar{\mathbf{p}} \end{bmatrix}, i_j \in N_v^k$$

$\mathbf{C}$  é uma matriz 3x3 que descreve as propriedades estatísticas de uma distribuição de pontos amostrais na vizinhança do ponto  $\mathbf{p}$ , pela acumulação dos quadrados das distâncias entre esses pontos e o centróide  $\bar{\mathbf{p}}$ . Considere o problema de autovetores de  $\mathbf{C}$

$$\mathbf{C} \cdot \mathbf{x}_l = \lambda_l \cdot \mathbf{x}_l, \quad l \in \{1,2,3\}$$

Como  $\mathbf{C}$  é simétrica e positiva semi-definida, pelo Teorema Espectral (Strang, 1998), todos os autovalores são reais e os autovetores  $\mathbf{x}_l$  formam uma base ortonormal, correspondendo às componentes principais do conjunto de pontos definido por  $N_v^k$ . Os autovalores  $\lambda_l$  medem a variação dos pontos em  $N_v^k$ , ao longo da direção do autovetor correspondente. A variação total, isto é a soma dos quadrados das distâncias entre os pontos e o centróide, é dada pelo traço de  $\mathbf{C}$ . Como o traço é invariante por transformações de similaridade ( $\mathbf{C} = \mathbf{A}^{-1} \mathbf{B} \mathbf{A}$ ), isto é, por mudança de base, então

$$\sum_{i \in N_v^k} |\mathbf{p}_i - \bar{\mathbf{p}}|^2 = \lambda_1 + \lambda_2 + \lambda_3$$

Assumindo que  $\lambda_1 \geq \lambda_2 \geq \lambda_3$ , segue-se que o plano definido pela equação

---

<sup>42</sup> A vizinhança pode ser obtida fazendo uma pesquisa em largura no grafo (Cormen, 2001) a partir do vértice  $v$ , interrompendo a pesquisa quando se alcança um nó com distância (em número de arestas do caminho desde  $v$ ) maior do que um número  $p$ .

$$(\xi - \bar{\mathbf{p}}) \cdot \mathbf{x}_3 = 0$$

passa por  $\bar{\mathbf{p}}$  e minimiza a soma dos quadrados das distâncias dos vizinhos de  $\mathbf{p}$  ( $\mathbf{x}_3$  é a direção de menor variância dos dados). Assim,  $\mathbf{x}_3$  é uma aproximação da normal à superfície em  $\mathbf{p}$ , ou em outras palavras,  $\mathbf{x}_1$  e  $\mathbf{x}_2$  geram o plano tangente em  $\mathbf{p}$ .

Na verdade, junto com essa avaliação da normal é necessário um indicador que avalie se a vizinhança de  $\mathbf{p}$  tem realmente uma estrutura planar. Os três autovalores da matriz de covariância correspondem à variância ao longo das direções principais. Assim, a relação entre seus valores informa sobre a estruturação da nuvem de pontos na vizinhança de  $\mathbf{p}$ : se os três autovalores são iguais ou aproximadamente iguais, a estrutura é isotrópica; se os dois menores autovalores são muito menores do que o maior, a estrutura é linear; no caso em que o menor autovalor é muito menor do que os demais, a estrutura é planar (Figura 5-7).

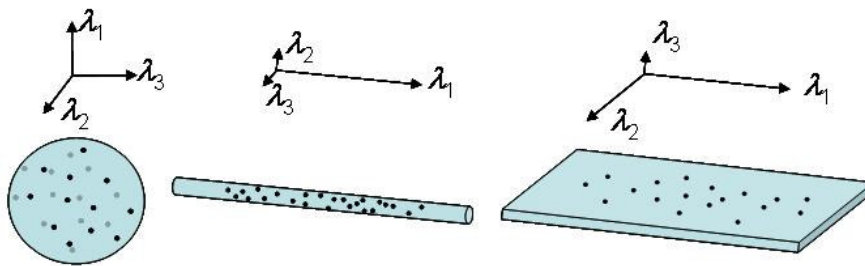


Figura 5-7: Estrutura de uma nuvem de pontos e sua relação com os autovalores da matriz de covariância. À esquerda, distribuição isotrópica; no centro, distribuição em linha; e, à direita distribuição planar.

No caso planar, o menor autovalor  $\lambda_3$  descreve quantitativamente o quanto a nuvem de pontos em torno de  $\mathbf{p}$  se desvia do plano tangente. A situação em que  $\lambda_3 = 0$  ocorre quando todos os pontos se distribuem exatamente sobre um plano. Nesse caso, pode-se afirmar que a estimativa do vetor normal é plenamente confiável. No caso em que  $\lambda_3 = \lambda_2$ , por outro lado, a nuvem de pontos se distribui sobre um cilindro. Nesse caso nada se pode afirmar sobre a direção da normal, e a informação dada pelo autovetor  $\mathbf{x}_3$  não tem valor como vetor normal. Na

prática não se deve comparar  $\lambda_3$  com zero, assim pode-se definir uma medida de confiança da estrutura como planar da seguinte forma<sup>43</sup>:

$$C_{plano} = \frac{\lambda_2 - \lambda_3}{\lambda_2 + \lambda_3}.$$

Raciocínio semelhante pode ser feito para o caso linear, gerando uma medida de confiança de estrutura linear:

$$C_{linha} = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}.$$

Na verdade, uma estrutura planar deve ocorrer apenas quando  $C_{plano} \approx 1$  e  $C_{linha} \approx 0$ . Assim, uma **medida de confiança da avaliação do vetor normal**  $C_{normal}$  deve levar em conta estas duas medidas. Nesta tese é utilizado:

$$C_{normal} = \begin{cases} C_{plano}, & \text{se } C_{linha} < \lim_{linha} \\ 0, & \text{caso contrário} \end{cases}$$

onde  $\lim_{linha}$  é um valor limite entre 0 e 1, acima do qual  $C_{linha}$  torna a estrutura linear.

Vale observar que o parâmetro de profundidade utilizado no cálculo da matriz de covariância funciona como um tamanho de filtro: uma profundidade com valor de 1 implica em avaliar a orientação média na vizinhança imediata do nó, uma profundidade maior implica em uma média sobre uma vizinhança larga.

#### 5.4.2. Segmentação por crescimento de regiões

O método apresentado na seção anterior permite, dada uma profundidade de vizinhança  $p$ , calcular para todos os nós do grafo sua matriz de covariância e obter, como resultado da análise de seus autovetores e autovalores, uma estimativa para a normal e a medida de confiança da estimativa. Tais informações irão guiar o algoritmo de crescimento de regiões.

O algoritmo trabalha com uma lista ordenada por confiança dos vértices do grafo. Essa é a lista de possíveis sementes. O nó de maior confiança é utilizado como semente inicial e o segmento cresce incluindo seus vértices vizinhos, ampliando a fronteira da região. Para que um vértice seja aceito na região é necessário que o ângulo entre a sua normal e a do vértice de fronteira que o

---

<sup>43</sup> As medidas de confiança planar e linear definidas aqui são semelhantes às propostas por Kempen et al. (1999) e Bakker (2002) para avaliação de orientação local pelo tensor de estrutura do gradiente (ver Apêndice A).



alcançou seja menor do que um limiar (parâmetro do algoritmo). Aqui está se assumindo explicitamente que as falhas são superfícies suaves. Na situação em que a fronteira da região não tem mais como avançar, uma nova semente é obtida da lista ordenada de vértices. Durante todo o algoritmo, são considerados apenas nós cuja confiança seja maior do que um limiar (parâmetro do algoritmo). Essa restrição se baseia no fato de que vértices com baixa confiança estão em regiões que não possuem estrutura planar compatível com uma superfície de falha (podem ser localmente compatíveis com uma linha ou com uma estrutura isotrópica). A seguir é apresentado o pseudo-código do algoritmo:

### Algoritmo de segmentação pela orientação local

A entrada é um grafo  $G = (V, A)$ , com  $n$  vértices e  $m$  arestas, a profundidade de vizinhança  $p$ , o parâmetro limiar do ângulo,  $limAng$ , e o parâmetro limiar da confiança,  $limConf$ . A saída é uma segmentação de  $V$  em componentes  $S = (S_1, \dots, S_r)$ .

1. Marcar todos os nós como disponíveis
2. Calcular para cada nó seu vetor normal e confiança  $C_{normal}$  em uma vizinhança de profundidade  $p$ .
3. Ordenar  $V$  em  $LConf = (o_1, \dots, o_k)$  em ordem decrescente de confiança  $C_{normal}$ . Não incluir em  $LConf$  nós com confiança  $\leq limConf$ .
4. Inicializar contador de segmentos, conjunto de nós do segmento e lista de nós da fronteira:  
 $r = 1; S_r = \emptyset; LFront = \emptyset;$
5. Retirar nó de  $LConf$  e inserir em  $LFront$ , marcando o nó como indisponível.
6. Enquanto (  $LFront$  não for vazia )
7.     Retirar nó  $s$  de  $LFront$  e inserir em  $S_r$ ;
8.     Para cada vizinho  $v$  de  $s$  faça
9.         Se (( $v$  está disponível) e (confiança de  $v > limConf$ ) e  
           (ângulo entre  $v, s < limAng$ )) então
10.             Inserir  $v$  em  $LFront$ , marcando o nó como  
               indisponível;
11.     Fim-se
12.     Fim-para
13.     Se (  $LFront$  está vazia ) então
14.         Executar procedimento ChecarSegmento (  $S_r$  )

15. Se ( $S_r$  é um segmento válido) então
16.  $S = S \cup \{ S_r \}; r = r+1;$
17. Fim-se
18.  $S_r = \emptyset;$
19. Percorrer  $LConf$  procurando por nó  $s$  disponível;
20. Se ( achou  $s$  ) então
21. Inserir  $s$  em  $LFront$ , marcando o nó como
22. indisponível.
23. Fim-se
24. Fim-se
25. Fim-enquanto.

Os passos de 1 a 5 correspondem à inicialização do algoritmo, com o cálculo dos vetores normais e suas confianças, a ordenação dos nós por confiança e a criação do primeiro segmento, inicialmente com apenas o nó de maior confiança. Observe também que os nós possuem um atributo de disponibilidade; isto é necessário devido ao fato de que um nó pode ser selecionado de duas maneiras diferentes: pode ser retirado da lista ordenada de nós ( $LConf$ ) ou pode ser retirado da lista de nós da fronteira da região ( $LFront$ ).

Os passos de 7 a 12 correspondem ao crescimento da região (segmento corrente). O passo 7 retira um nó  $s$  disponível da fronteira e o insere na região. Os passos 8 a 12 inserem os vizinhos disponíveis do nó  $s$  na fronteira. Isto acontece apenas no caso em que o ângulo entre os vetores normais de  $s$  e seu vizinho é menor do que  $limAng$  e a confiança do vizinho é maior do que  $limConf$ .

Os passos 13 a 24 tratam da finalização da região corrente e inicialização da próxima. O passo 13 testa se a fronteira está vazia, o que significa que a região (segmento) está completa. Em caso positivo, o procedimento ChecarSegmento verifica se o segmento obtido satisfaz outras restrições. Caso positivo, o segmento é incluído no conjunto de segmentos. O trecho de código continua procurando na lista ordenada de nós por um nó que inicie um novo segmento.

O segmento obtido por esse método de segmentação deve corresponder a uma superfície suave. Entretanto, falta impor a condição de verticalidade para a superfície de falha. Ela pode ser incluída no procedimento ChecarSegmento do algoritmo acima, estabelecendo um percentual mínimo de nós com mergulho maior do que um parâmetro. Outra restrição seria limitar o número mínimo de nós para um segmento aceitável, descartando regiões muito pequenas.

A seguir é apresentada uma tabela com os parâmetros do algoritmo de segmentação por orientação local, seus respectivos significados e valores típicos:

Parâmetro	Descrição	Valores típicos
$p$	Profundidade para pesquisa da vizinhança	2
$limAng$	Ângulo limite entre nós vizinhos	0.3 rad
$limConf$	Confiança de normal mínima	0.45

Tabela 3: Parâmetros da segmentação por orientação.

### 5.4.3. Resultados

A seguir são apresentadas algumas Figuras com resultados da segmentação pela orientação. Foi utilizado como dado de entrada o grafo segmentado pelo algoritmo FH com  $k=15$  (ver Figura 5-4). Em todos os casos foram utilizados  $p=2$ ,  $limAng=0.3rad$  ( $17^\circ$ ) e  $limConf=0.45$ . O tempo de execução desta etapa de segmentação é da ordem de 8s, enquanto que a do algoritmo FH foi de menos de 1s. Estes valores de tempo foram obtidos com a utilização de equipamento com cpu Pentium 4 de 3.4GHz.

Na Figura 5-8 é apresentado o resultado da segmentação sem aplicar a restrição de verticalidade e número mínimo de vértices. Observe que a região não vertical de aspecto caótico do segmento vermelho da Figura 5-4 (ver também Figura 5-6) foi eliminado, sendo separado em um conjunto de segmentos pequenos. As três grandes estruturas estão preservadas.

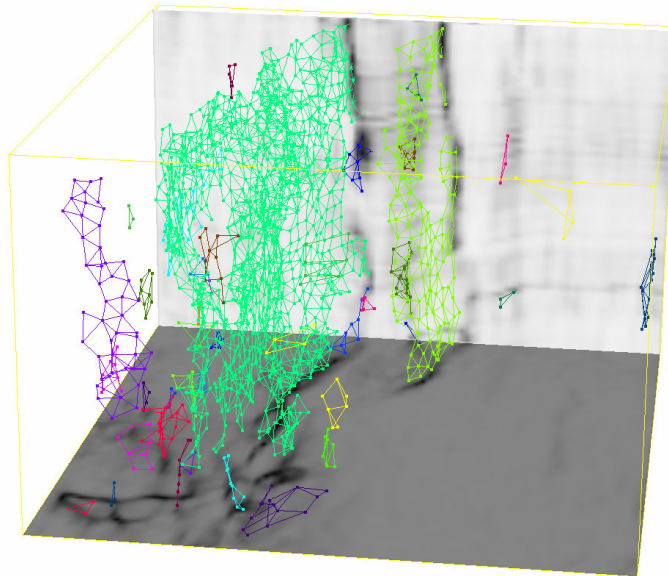


Figura 5-8: Resultado da segmentação pela orientação sem imposição de verticalidade. Comparar com a Figura 5-4.

Da Figura 5-9 em diante foi aplicada a restrição de verticalidade, exigindo que pelo menos um nó do segmento tenha um mergulho maior do que 1 rad ( $57^\circ$ ) e que o número de nós seja maior ou igual a 40. As Figuras exibem fatias verticais do atributo de falha sobrepostas ao grafo, mostrando a adequação do grafo resultante dos dois passos de segmentação com as principais estruturas de falha.

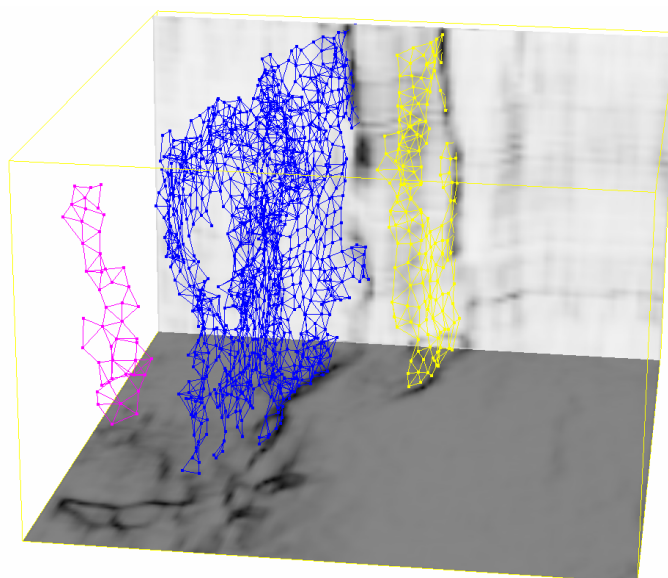


Figura 5-9: Resultado da segmentação pela orientação, impondo verticalidade e número de nós maior do que 40, por segmento.

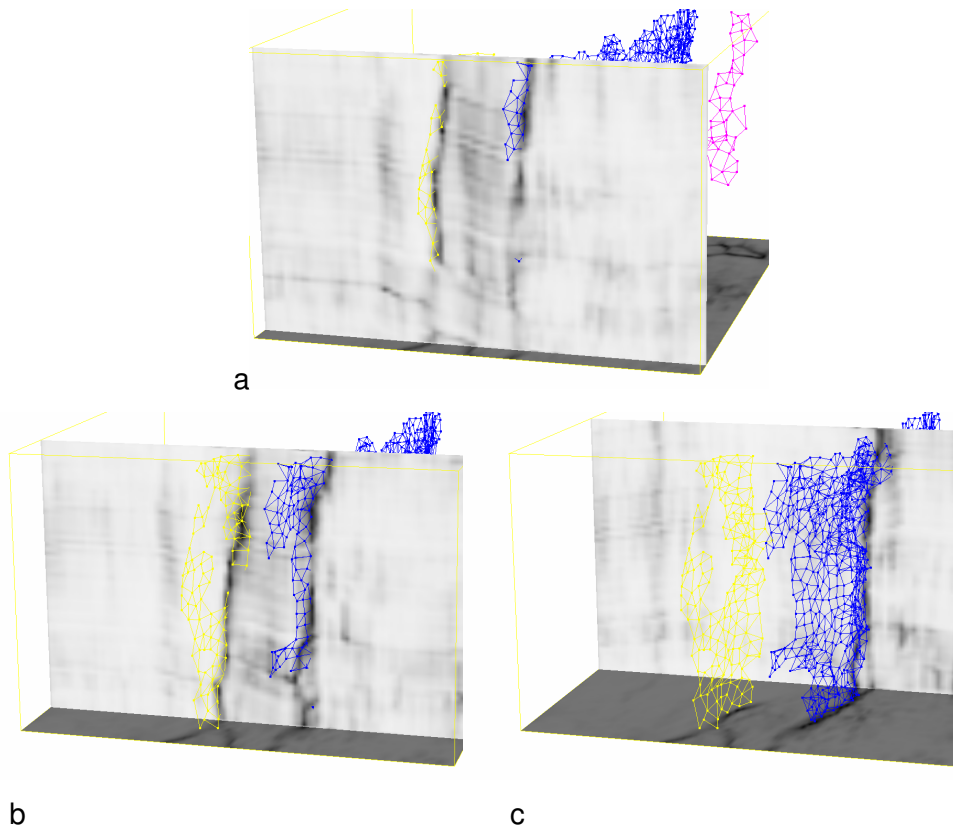


Figura 5-10: Fatias ao longo de uma mesma direção vertical do atributo de falhas em 3 posições diferentes, mostrando como o grafo resultante da segmentação acompanha as principais estruturas de falha.

Observe que as regiões onde o grafo não está presente, em geral, correspondem a regiões do atributo de falha com valores muito baixos (por exemplo, Figura 5-11b) ou onde o atributo se mostra localmente caótico, isto é não apresenta localmente uma estrutura próxima à de uma superfície (por exemplo, Figura 5-11c).

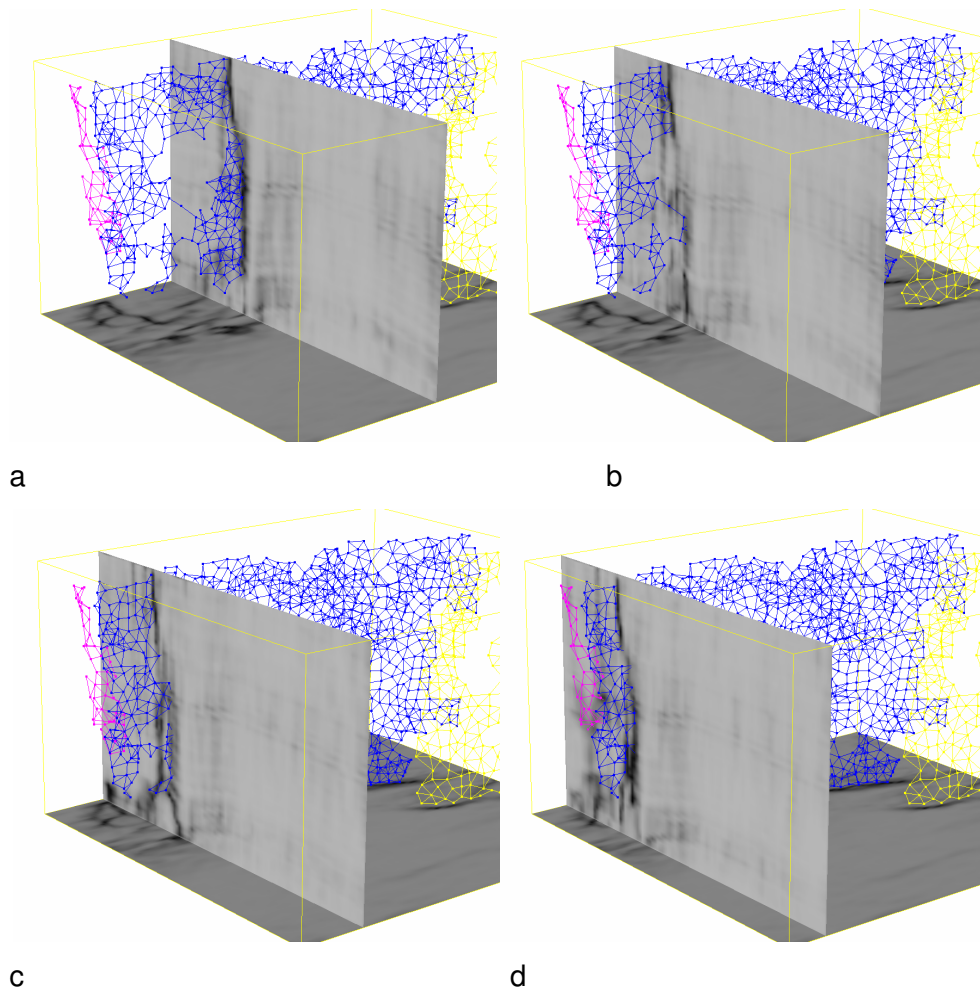


Figura 5-11: Fatias ao longo de uma mesma direção vertical (mesma direção da Figura anterior) do atributo de falhas em 4 posições diferentes, mostrando como o grafo resultante da segmentação acompanha as principais estruturas de falha.

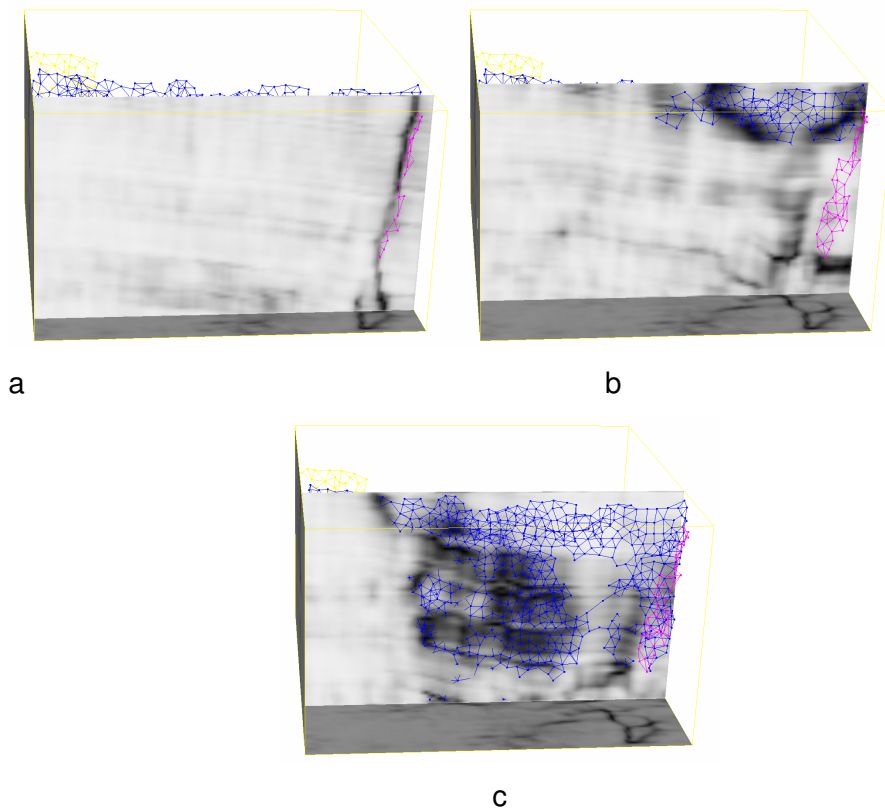


Figura 5-12: Fatias ao longo de uma mesma direção vertical (direção perpendicular à da Figura anterior) do atributo de falhas em 3 posições diferentes, mostrando como o grafo resultante da segmentação acompanha as principais estruturas de falha.