Solução de sistemas de equações não-lineares pelo método do recozimento simulado

Eloy Kaviski
Depto de Hidráulica e Saneamento, UFPR
81531-990, Curitiba, PR
E-mail: eloy.dhs@ufpr.br

Adriana Luiza do Prado
Depto de Matemática, UFPR
81531-990, Curitiba, PR
E-mail: alprado@ufpr.br

Liliana Madalena Gramani Cumin
Depto de Matemática, UFPR
81531-990, Curitiba, PR
E-mail: gramani@mat.ufpr.br

Na solução de problemas pelo método dos elementos finitos geralmente são definidas equações não-lineares onde as incógnitas são os deslocamentos nodais do sistema discretizado [8]. Esses sistemas de equações podem ser representados por:

$$N(d) = F \qquad (1)$$

Sendo N a matriz de rigidez da estutura, d é o vetor dos deslocamentos nodais, F é o vetor das forças exteriores.

O sistema de equações (1) pode ser solucionado pelo método de Newton-Raphson, e pela técnica *arc-length* que introduz uma variação da carga durante o processo iterativo correspondente ao método de Newton-Raphson [2]. O nível da carga passa a ser também uma incógnita e torna-se necessário considerar uma equação adicional. Esta equação restringe a solução de forma a cumprir um determinado critério. Na Figura 1 apresenta-se a resposta de uma estrutura com comportamento complexo com procedimentos de *load control* (incremento de carga ΔF) e *displacement control* (incremento de deslocamento Δa), para que as respostas entre todos os pontos seja encontrada. Quando a resposta entre os pontos entre A e D não são obtidas, i.e., a resposta é constituída apenas pelos pontos situados sobre a curva entre O e A e pelos pontos a partir de D este fenômeno é conhecido por *snap-through*. Do mesmo modo quando não são obtidos os pontos situados sobre a curva entre B e C, i.e., a resposta é constituída apenas pelos pontos O a B e pelos pontos a partir de C. Este fenômeno é conhecido por *snap-back*. Nestas condições os métodos passam a ser designados métodos com solução restringida.

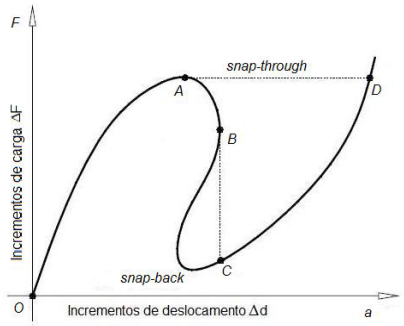

Figura 1. Resposta F-a de uma estrutura com os fenômenos de *snap-through* e *snap-back*.

Neste trabalho apresenta-se a solução do sistema de equações (1) pelo método de recozimento simulado (*simulated annealing*) [3]. O método do recozimento simulado é uma técnica que recentemente tem recebido muita atenção para solucionar problemas de otimização de grande porte [1,6]. Este método foi concebido por analogia com o algoritmo de Metropolis [7], que foi proposto para simular pelo método de Monte Carlo problemas de física estatística [5,8].

O uso do algoritmo de Metropolis pode ser descrito considerando-se um sistema cujos

estados são não degenerados com energias $E1<E2<E3<...<En$. Pelo algoritmo de Metropolis, o objetivo do problema consiste em simular a evolução do sistema em equilíbrio térmico que encontra-se na temperatura T. Considera-se inicialmente que o sistema encontra-se no estado j (energia Ej). O sistema poderá transitar para qualquer um dos outros estados f. A viabilidade da mudança de estado é analisada calculando-se o acréscimo de energia $\Delta E=Ej-Ef$ e o fator de Boltzmann $\exp[-\Delta E/(kT)]$, sendo k a constante de Boltzmann ($1{,}380658/10^{23}$ J/K). Gera-se um número aleatório uniforme p ($0\leq p\leq 1$) e compara-se com o fator de Boltzmann: (i) se $p>\exp[-\Delta E/(kT)]$, a proposta é recusada, permanecendo o sistema no estado j;
(ii) se $p\leq\exp[-\Delta E/(kT)]$, a proposta é aceita, e o sistema será alterado para o estado f.

Como conseqüência, se $\Delta E<0$, ou seja ao transitar é diminuída a energia do sistema, a proposta é sempre aceita; e se $\Delta E>0$, a proposta pode ser aceita com probabilidade p, e caso seja aceita, significa que aumentará a energia do sistema, sendo este o aspecto, um pouco diferente da idéia que um sistema tende sempre a diminuir a sua energia, que fortalece e acrescenta importância ao algoritmo de Metropolis.

Para fazer uso do algoritmo de Metropolis para solucionar problemas gerais de otimização, devem ser conhecidos os seguintes elementos [8]:
(i) descrição das possíveis configurações do sistema;
(ii) um gerador aleatório para realizar mudanças na configuração; estas mudanças são opções que devem fazer parte das configurações do sistema;
(iii) uma função objetivo E (analogia com energia) cuja minimização é o objetivo do procedimento;
(iv) um parâmetro de controle (analogia com temperatura) e um esquema de recozimento para que o sistema possa ser conduzido para o menor valor da função objetivo.

Para atender os propósitos deste trabalho foi desenvolvido um método computacional onde a função objetivo foi definida como o valor absoluto máximo selecionado entre os resíduos das equações do sistma de equações analisado. As variáveis de decisão são as incógnitas do sistema de equações.

A solução inicial é obtida através de amostragem aleatória obtida na região de busca da solução. De maneira independente, para cada um dos componentes do vetor de variáveis de decisão, são gerados valores uniformente distribuídos compreendidos nos seus respectivos limites de validade. Na seqüência verifica-se através das equações de restrições a viabilidade da solução. O processo é repetido até que consiga-se obter uma solução aleatória viável.

O algoritmo de Metropolis é usado para gerar um conjunto de pontos num espaço de variáveis distribuídas com função densidade de probabilidade estimadas pelo fator de Boltzmann. O parãmetro kT inicialmente é definido como 0.5, sendo posteriormente reduzido por um fator igual a 0.9, em cada um dos 100 passos considerados. Desta forma, gera-se uma seqüência de pontos (identificados genericamente pelo vetor $\boldsymbol{X}$, cujos componentes são as variáveis de decisão) representando um caminho aleatório movendo-se através do espaço configurado.

As regras pelas quais o caminho aleatório é realizado no espaço são as seguintes:
(i) considera-se que o caminho aleatório encontra-se no ponto $\boldsymbol{X}_n$;
(ii) para gerar o ponto $\boldsymbol{X}_{n+1}$ aplica-se um processo iterativo. O novo ponto pode ser escolhido aleatoriamente sobre a superfície de uma hiperesfera com raio de pequena dimensão δ, em torno do ponto $\boldsymbol{X}_n$;
(iii) sorteando-se um possível ponto $\boldsymbol{X}_t$, esta solução é aceita ou rejeitada considerando-se a razão:

$$r=\exp[-\Delta E/(kT)], \qquad (2)$$

sendo $\Delta E=E(\boldsymbol{X}_n)-E(\boldsymbol{X}_{n+1})$, onde $E(\boldsymbol{X}_n)$ é o valor da função objetivo no ponto $\boldsymbol{X}_n$;
(iv) Se $r>1$, então o ponto $\boldsymbol{X}_t$ é aceito ($\boldsymbol{X}_{n+1}=\boldsymbol{X}_t$), enquanto que se $r<1$, o ponto $\boldsymbol{X}_t$ é aceito com probabilidade r. Este procedimento é realizado comparando-se r com um número u

uniformemente distribuído no intervalo [0,1], aceitando-se $\boldsymbol{X}_t$, se $u<r$. Quando o ponto $\boldsymbol{X}_t$ não é aceito o caminho aleatório permanece no ponto $\boldsymbol{X}_n$ ($\boldsymbol{X}_{n+1}=\boldsymbol{X}_n$); e (v) gera-se o ponto $\boldsymbol{X}_{n+2}$ usando-se o mesmo procedimento.

O valor de δ deve ser escolhido de forma que 1/3 a 1/2 das configurações geradas sejam aceitas, caso contrário o método torna-se pouco eficiente [4]. Se houver uma grande quantidade de configurações rejeitadas significa que o valor de δ é muito grande; caso contrário se δ é muito pequeno, há muitas configurações aceitas mas a região explorada pelo método é pequena. A melhor escolha para o ponto inicial $\boldsymbol{X}_0$ encontra-se onde a distribuição de probabilidades corresponda a um máximo.

O método descrito foi aplicado para solucionar o seguinte problema:

$$N(d)=\begin{Bmatrix}10d_1+0{,}4d_2^2-5d_2^3\\0{,}4d_1^3-3d_1^2+10d_2\end{Bmatrix};F=\begin{Bmatrix}40t\\15t\end{Bmatrix} \quad (3)$$

Sendo t um parâmetro pertencente ao intervalo $[0,2]$. O problema não-linear a ser resolvido é o de equilíbrio de forças:

$$R(d_{n+1})=F(t_{n+1})-F(d_{n+1}) \therefore$$

$$R(d_{n+1})=\begin{Bmatrix}40t-10d_1+0{,}4d_2^2-5d_2^3\\15t-0{,}4d_1^3-3d_1^2+10d_2\end{Bmatrix} \quad (4)$$

Na Tabela 1 apresenta-se os resultados obtidos pelo método do Recozimento Simulado. Na Figura 1 apresenta-se o resultado obtido com o método de Newton-Raphson com Comprimento de Arco usando uma carga b=0.

t	d1	d2
0.00	0.0000000	0.0000000
0.00	3.4557196	1.9318730
0.00	5.7953785	2.2900843
0.05	0.2000256	0.0866824
0.05	3.2161803	1.8474428
0.05	5.9322810	2.2818418
0.10	0.4022253	0.1959318
0.10	2.9790654	1.7548988
0.10	6.0521715	2.2712906
0.15	0.6134300	0.3286556
0.15	2.7323916	1.6487918
0.15	6.1598585	2.2590071
0.20	0.8499136	0.4921495
0.20	2.4576754	1.5182600
0.20	6.2582811	2.2453325
0.25	1.1651826	0.7190197
0.25	2.1007522	1.3281090
0.25	6.3493713	2.2304810
0.30	6.4344917	2.2145956
0.35	6.5146366	2.1977727
0.40	6.5905610	2.1800777
0.45	6.6628541	2.1615564
0.50	6.7319900	2.1422371
0.55	6.7983523	2.1221380
0.60	6.8622602	2.1012649
0.65	6.9239762	2.0796166
0.70	6.9837332	2.0571849
0.75	7.0417272	2.0339547
0.80	7.0981294	2.0099040
0.85	7.1530934	1.9850045
0.90	7.2067503	1.9592230
0.95	7.2592269	1.9325174
1.00	7.3106297	1.9048376
1.05	7.3610641	1.8761298
1.10	7.4106285	1.8463249
1.15	7.4594080	1.8153472
1.20	7.5074928	1.7831078
1.25	7.5549681	1.7495019
1.30	7.6019192	1.7144076
1.35	7.6484296	1.6776819
1.40	7.6945914	1.6391551
1.45	7.7404963	1.5986237
1.50	7.7862491	1.5558431
1.55	7.8319591	1.5105110
1.60	7.8777622	1.4622551
1.65	7.9238141	1.4106020
1.70	7.9703054	1.3549393
1.75	8.0174792	1.2944562
1.80	8.0656614	1.2280369
1.85	8.1153033	1.1540917
1.90	8.1670751	1.0702136
1.95	8.2220521	0.9725077
2.00	8.2821640	0.8539222

Tabela 1: Resultados obtidos pelo método do Recozimento Simulado

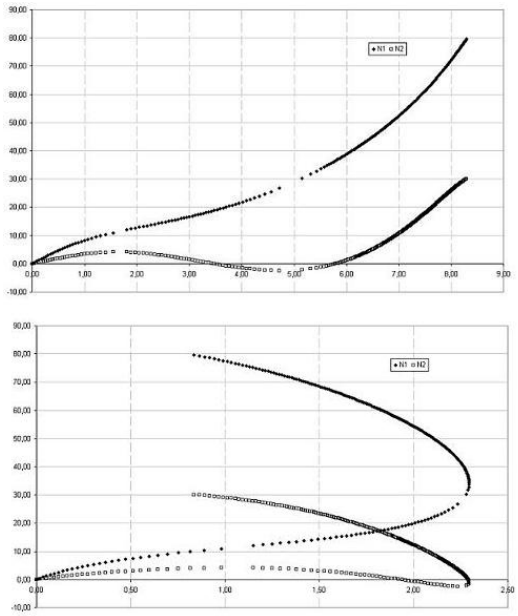

Figura 1

Ao utilizar o método do Recozimento Simulado para $t < 3$ existem 3 soluções no intervalo de busca, para $t \geq 3$ existe somente uma solução no intervalo de busca. A solução positiva encontrada é continuação da terceira solução. Observe que se considerando a terceira solução não existe um salto na solução. Avaliando os gráficos do método de Newton-Raphson com Comprimento de Arco (carga b=0) observa-se que existem pontos onde d_1 e d_2 não são encontrados. Esta deficiência do método de Newton-Raphson na busca da solução deste problema não ocorre quando é utilizado o método do Recozimento Simulado. Na Figura 2 apresenta-se o algoritmo do método do Recozimento Simulado implementado em Borland Pascal.

```
Listagem do programa rec_sim.
program rec_sim;
  type
    pec  = array[1..2] of double;
    vpec = array[1..5] of pec;
    eab  = ^ab;
    ab   = record
             a,b : double;
             p : eab;
             end;
  var
    arq    : text;
    ctt    : pec;
    a2,b2,tt : double;
    abi,ab1,ab2 : eab;
    ns,nsa,isa,i  : byte;
    s,sa   : ^vpec;
function sqd(var z,x:pec):double;
{ Função objetivo: Resíduos das equações }
    var
      x12,x22,
      aux,bux : double;
    begin
      x[1] := ab1^.a + ab1^.b*z[1];
      x[2] := a2 + b2*z[2];
      x12 := x[1]*x[1];
      x22 := x[2]*x[2];
      aux := abs(10.0e0*x[1] + x[2]*
             (0.4e0*x[2]-5.0*x22)-ctt[1]);
      bux := abs(10.0e0*x[2] + x[1]*
             (0.4e0*x12-3.0*x[1])-ctt[2]);
      if aux > bux
        then sqd := aux
        else sqd := bux;
      end;
procedure otimo;
{ Solução do sistema de equações.
  ran3  - gera n. aleatórios uniformes [0,1].
  norbm - gera n. aleatórios normais com
          média=0 e variância=1. }
  var
    i,j,nt : word;
    nsim,nsuc,simmax,sucmax : longint;
    df,fat,t,fmin,fxr,fyr  : double;
    xmin     : pec;
    xr,yr,zr : ^pec;
    fora     : boolean;
  const
    eps : double = 1.0e-5;
procedure sorte;
{ Determina solução candidata }
  var
    aux,r : double;
    i     : byte;
const
    rc : double = 1.0e-3;
  begin
    repeat
      r := 0.0e0;
      for i := 1 to 2 do
        begin
          zr^[i] := norbm;
          r      := r + zr^[i]*zr^[i];
          end;
      until r > 1.0e-16;
    r := ran3*rc/sqrt(r);
   for i := 1 to 2 do
      begin
        aux    := r*zr^[i];
        yr^[i] := xr^[i] + aux;
        if yr^[i] < 0.0e0
          then yr^[i] := 0.0e0
          else if yr^[i] > 1.0e0
            then yr^[i] := 1.0e0;
        end;
    end;
function metropolis:boolean;
  var
    aux : double;
  begin
    metropolis := true;
    if df < 0.0
      then exit;
    aux := df/t;
```

```
    if aux < 200.0
      then if ran3 < exp(-aux)
        then exit;
    metropolis := false;
    end;
procedure novo_inter;
  var
    aux,bux : double;
  const
    err : double = 0.1e0;
  begin
    if tt > eps
      then exit;
    aux := xmin[1] - err;
    if aux > ab1^.a
      then
        begin
          new(ab2^.p);
          ab2    := ab2^.p;
          ab2^.p := nil;
          ab2^.a := ab1^.a;
          ab2^.b := aux - ab2^.a;
          end;
    aux := xmin[1] + err;
    bux := ab1^.a + ab1^.b;
    if aux < bux
      then
        begin
          new(ab2^.p);
          ab2    := ab2^.p;
          ab2^.p := nil;
          ab2^.a := aux;
          ab2^.b := bux - ab2^.a;
          end;
    end;
  begin { otimo }
    new(xr); new(yr); new(zr);
    ctt[1] := 40.0e0*tt;
    ctt[2] := 15.0e0*tt;
    simmax :=   5000;
    sucmax :=   1000;
    nt     :=    100;
    fat    := 0.90e0;
    t      := 0.50e0;
    if tt < eps
      then
        begin
          fxr := 1.0e50;
          for i := 0 to 100 do
            begin
              yr^[1] := 0.01e0*i;
              for j := 0 to 100 do
                begin
                  yr^[2] := 0.01e0*j;
                  fyr := sqd(yr^,zr^);
                  if fyr < fxr
                    then
                      begin
                        xr^  := yr^;
                        fxr  := fyr;
                        xmin := zr^;
                        fmin := fxr;
                        end;
                  end;
              end;
          end
      else begin
        isa := isa + 1;
        if isa > nsa
          then begin
           ab1^.p := nil;
             exit;
             end
          else ab1^.p := ab1;
        xr^[1] := (sa^[isa][1]-
ab1^.a)/ab1^.b;
        xr^[2] := (sa^[isa][2]-a2)/b2;
        fxr  := sqd(xr^,xmin);
        fmin := fxr;
        end;
      for j := 1 to nt do
        begin
          nsuc := 0;
          nsim := 0;
          repeat
            nsim := nsim + 1;
            sorte;
            fyr := sqd(yr^,zr^);
            df  := fyr - fxr;
            if metropolis
              then begin
                nsuc := nsuc + 1;
                xr^  := yr^;
                fxr  := fyr;
                if fxr < fmin
                  then begin
                    xmin := zr^;
                    fmin := fxr;
                   if fmin < eps
                      then begin
                        ns := ns + 1;
                        s^[ns] := xmin;
                        novo_inter;
                        dispose(xr);
                        dispose(yr);
                        dispose(zr);
                        exit;
                        end;
                    end;
                end;
            until (nsim>simmax) or
(nsuc>sucmax);
          t := t*fat;
          end;
      if fmin < 0.1
        then begin
          ns := ns + 1;
          s^[ns] := xmin;
          novo_inter;
          end;
      dispose(xr); dispose(yr); dispose(zr);
      end;
    begin { rec_sim }
      new(s); new(sa); new(abi);
      a2 := 0.0e0;
      b2 := 1.0e1;
      abi^.a := a2;
      abi^.b := b2;
      abi^.p := nil;
      assign (arq,'rec_sim.txt');
      rewrite(arq);
      tt := 0.0e0;
      repeat
        ns  := 0;
        isa := 0;
        ab1 := abi;
        ab2 := abi;
        repeat
          otimo;
          ab1 := ab1^.p;
          until ab1 = nil;
        for i := 1 to ns do
```

```
writeln(arq,tt:8:3,s^[i][1],s^[i][2]);
      nsa := ns;
      sa^ := s^;
      ab1    := abi^.p;
      abi^.p := nil;
      while ab1 <> nil do
        begin
          ab2 := ab1;
          ab1 := ab1^.p;
          dispose(ab2);
          end;
      tt := tt + 0.05e0;
      until tt > 2.05e0;
    close(arq);
    end.
```

Figura 2: Listagem do Programa do método do Recozimento Simulado

Referências

[1] M. C. Cunha, J. Souza, Hydraulic infrastructures design using simulated annealing. Journal of Infrastructure Systems, 7 (1), 32-39, (2001).

[2] V. Gouveia, J. Barros, A. Azevedo, J. S. Cruz, Implementação da técnica do arc-length e métodos relacionados no programa de lementos finitos FEMIX, Relatório 06-DEC/E-20, Universidade do Minho, Instituto Politécnico de Viseu, Universidade do Porto, 2005.

[3] E. Kaviski, L. M. da Cunha, D. Lambros, C. S. Garcia. Parametrização de curvas de intensidade de precipitação pelo método de recozimento simulado, XVII Simpósio Brasileiro de Recursos Hídricos, 25-29/11/2007, São Paulo, (2007)

[4] S. E. Koonin, D. C. Meredith, Computational Physics - Fortran Version, Addison-Wesley, 1993.

[5] E. J. S. Lage, Física Estatística, Fundação Calouste Gulbenkian, Lisboa, 1995.

[6] A. R. Mckendall, J. Shang, S. Kuppusamy, Simulated annealing heuristics for the dynamic facility layout problem, Computer & Operations Research, 33, 2431-2444 (2006).

[7] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, Equation of state calculations by fast computing machines, J. Chem.Phys. 21 (6), 1087 (1953).

[8] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, Numerical Recipes in Pascal, Cambridge, 1992.

[9] O. C. Zinkiewicz, K. Morgan, Finite Elements and Aproximation, J. Wiley, 1983.