ỨNG DỤNG CÁC THUẬT TOÁN CĂN BẬC HAI VÀO TÍNH TOÁN BÌNH SAI TRUY HỒI LƯỚI TRẮC ĐỊA

NGUYỄN NGỌC LÂU[(1)], NGUYỄN THỊ THANH HƯƠNG[(2)]
[(1)]Trường Đại học Bách khoa TP. HCM
[(2)]Viện Khoa học Đo đạc và Bản đồ

Tóm tắt:

Để bình sai lưới trắc địa một cách hiệu quả bằng máy tính, người ta thường ứng dụng các thuật toán bình sai truy hồi (Recursive/Sequential Adjustment). Trong đó thường được sử dụng nhiều nhất là thuật toán Q. Thuật toán Q duy trì việc tính lặp trên ma trận phương sai đối xứng, được đánh giá là đôi khi gặp khó khăn trong các tính toán số và làm giảm độ chính xác của kết quả bình sai. Trong bài báo này, chúng tôi đã tìm hiểu các thuật toán căn bậc hai có độ ổn định tính toán số tốt hơn và áp dụng chúng vào việc bình sai lưới trắc địa. Dựa trên các thuật toán dẫn ra, chúng tôi đã thiết kế các chương trình MATLAB sao cho có thể tiết kiệm bộ nhớ nhất, và dùng ví dụ minh họa về lưới thủy chuẩn để chứng minh tính đúng đắn của chúng.

1. Giới thiệu:

Bình sai truy hồi (Recursive/Sequential Adjustment) đã được giới thiệu từ thập niên 70-80 để xử lý các mạng lưới trắc địa [1, 2, 3, 4] với nhiều ưu điểm khi so với phương pháp bình sai cổ điển như:

* Tính toán hiệu quả trên máy tính điện tử (bộ nhớ và tốc độ)

* Có khả năng tích hợp kiểm tra sai số thô vào quá trình tính toán

Nội dung của phương pháp là tính toán lặp dần trên các yếu tố:

* Ma trận hiệp phương sai (variance-covariance matrix) của vector ẩn số Q

* Vector ẩn số X

* Dạng toàn phương ϕ

theo từng trị đo được đưa vào quá trình xử lý. Do thuật toán tính lặp trên ma trận Q, để thuận tiện ta gọi là thuật toán Q, có thể tóm tắt như sau [4]:

Bước 1: Bắt đầu với ma trận hiệp phương sai của vector ẩn số $Q_{(0)} = 10^6 E$, giá trị ban đầu của vector ẩn số $\hat{X}_{(0)}$ và dạng toàn phương $\hat{\phi}_{(0)} = 0$

i = 1

Bước 2: Tính số hạng tự do của trị đo thứ i

$$w_i = f_i\left(\hat{X}_{(i-1)}\right) - y_i \qquad (1)$$

Trọng số đảo của số hạng tự do w_i

$$q_{w_i} = \frac{1}{p_i} + a_i Q_{(i-1)} a_i^T \qquad (2)$$

Kiểm tra $|w_i| < 3\sigma_0\sqrt{q_{w_i}}$ (3)

Ngày nhận bài: 12/2/2022, ngày chuyển phản biện: 15/2/2022, ngày chấp nhận phản biện: 19/2/2022, ngày chấp nhận đăng: 25/2/2022

Nếu (3) không thỏa mãn chuyển sang bước 3

Ước lượng của ẩn số sau khi xử lý trị đo thứ i

$$\hat{X}_{(i)} = \hat{X}_{(i-1)} - Q_{(i-1)}a_i^T \frac{w_i}{q_{w_i}} \quad (4)$$

Ma trận hiệp phương sai của ẩn số sau khi xử lý trị đo thứ i

$$Q_{(i)} = Q_{(i-1)} - \frac{Q_{(i-1)}a_i^T a_i Q_{(i-1)}}{q_{w_i}} \quad (5)$$

Dạng toàn phương sau khi xử lý trị đo thứ i

$$\hat{\phi}_{(i)} = \hat{\phi}_{(i-1)} + \frac{w_i^2}{q_{w_i}} \quad (6)$$

Bước 3: i = i +1

nếu i > n thì chuyển sang bước 4, ngược lại quay về bước 2

Bước 4: Xuất kết quả bình sai

Trong đó: y_i là trị đo thứ i, có trọng số p_i (i = 1:n)

a_i là vector hệ số của phương trình tham số của y_i

Các ma trận hay vetor trong thuật toán trên có chỉ số nằm trong ngoặc để chỉ định ma trận hay vector đó được khai báo duy nhất trong chương trình, chỉ có giá trị của chúng là thay đổi theo bước tính lặp.

Krakiwsky [1] đã chứng minh bình sai truy hồi là một trường hợp riêng của Kalman filter khi vector trạng thái không thay đổi theo thời gian. Trong bài báo của mình, Kalman đã đặt:

$$K_i = \frac{Q_{(i-1)}a_i^T}{q_{w_i}} \quad (7)$$

và gọi là ma trận gia lượng (gain matrix). Khi đó các công thức truy hồi (4) và (5) có thể viết lại ở dạng

$$\hat{X}_{(i)} = \hat{X}_{(i-1)} - K_i w_i \quad (8)$$

$$Q_{(i)} = Q_{(i-1)} - K_i a_i Q_{(i-1)} \quad (9)$$

Mặc dù thuật toán Q rời rạc đã được khai thác thành công trong nhiều bài toán, đôi khi các ứng dụng thực tế thường đối diện với những khó khăn về tính toán số. Một vài nhà nghiên cứu đã thông báo về việc giảm độ chính xác của thuật toán này. Các khó khăn tính toán với phương pháp Kalman thường thấy rõ rệt nhất trong trường hợp các ma trận phương sai đã tính được trở nên không xác định (undefine) hoặc ở điều kiện xấu (ill condition). Việc giảm đi tính xác định của ma trận hiệp phương sai thường là kết quả sai số làm tròn của máy tính, đã làm trầm trọng hơn bởi điều kiện xấu. Những trường hợp xấu có thể xảy ra khi các trị đo rất chính xác được xử lý liên kết với các phương sai ban đầu lớn, hay khi một tổ hợp tuyến tính của các tham số có thể được ước lượng một cách chính xác trong khi những tham số khác thì không. Trong những trường hợp này các tính toán liên quan với ma trận hiệp phương sai dễ mắc phải sai số làm tròn.

2. Các thuật toán căn bậc hai

Nhiều sơ đồ tính toán đã được đề xuất để khắc phục việc giảm tính xác định của ma trận phương sai. Những phương pháp này hoàn toàn tương đương về mặt đại số với thuật toán Q, chúng đại diện cho những kỹ thuật tính toán khác được thiết kế để cải thiện độ chính xác số. Các thuật toán này thường đòi hỏi khối lượng tính toán lớn hơn công thức ban đầu, và vài phương pháp còn đòi hỏi việc lưu trữ nhiều hơn trên máy tính. Do đó, nhiều phương pháp không thích hợp cho những ứng dụng có thời gian và khả năng lưu trữ của máy tính giới hạn. Tình huống này thường gặp trong nhiều

ứng dụng thời gian thực, như định vị on-board của máy bay và tên lửa.

Việc lập công thức cho ma trận hiệp phương sai ở dạng căn bậc hai đã được nhiều nhà nghiên cứu công nhận là tốt hơn về mặt tính toán số so với công thức cổ điển. Sau đây chúng tôi xin giới thiệu hai thuật toán được đánh giá cao nhất trong nhóm này là thuật toán Carlson và thuật toán U-D.

2.1. Thuật toán căn bậc hai tam giác của Carlson

Với mong muốn đạt được sự ổn định và các đặc tính chính xác của kỹ thuật ước lượng căn bậc hai và sự cần thiết cho một thuật toán xử lý nhanh và tin cậy, Carlson vào năm 1973 đã đề xuất thay thế bằng một công thức căn bậc hai. Phương pháp của ông ta tính toán truy hồi căn bậc hai một ma trận hiệp phương sai có dạng tam giác trên [3]. Dựa vào ý tưởng của Carlson, chúng tôi tự chứng minh và dẫn dắt ra các công thức cần thiết ở phần sau. Do đó chúng có thể không hoàn toàn giống với các công thức Carlson nguyên bản

Vì Q là ma trận đối xứng xác định dương nên ta có thể phân tích thành

$$Q = U.U^T \qquad (10)$$

Trong đó U là ma trận tam giác trên

Kết hợp (5) và (10), ta có

$$U_{(i)}U_{(i)}^T = U_{(i-1)}S_iU_{(i-1)}^T \qquad (11)$$

Trong đó

$$S_i = E - \frac{1}{q_{w_i}}t_it_i^T \qquad (12)$$

$$t_i = U_{(i-1)}^T a_i^T \qquad (13)$$

Vì Q_i là ma trận đối xứng và xác định dương nên S_i cũng đối xứng và xác định dương. Do đó tồn tại duy nhất các ma trận tam giác trên U sao cho

$$S_i = \bar{U}_i\bar{U}_i^T \qquad (14)$$

Thay (14) vào (11), ta được

$$U_{(i)}U_{(i)}^T = U_{(i-1)}\bar{U}_i\bar{U}_i^TU_{(i-1)}^T \qquad (15)$$

Suy ra $U_{(i)} = U_{(i-1)}\bar{U}_i$ (16a)

Hay $U_{(i)}^T = \bar{U}_i^TU_{(i-1)}^T$ (16b)

Vậy vấn đề còn lại là làm sao thành lập được ma trận $\bar{U}$ một cách hiệu quả nhất? Dựa vào thuật toán Cholesky, ta có thể dẫn ra các công thức khá đơn giản sau đây để thành lập ma trận $\bar{U}$ một cách trực tiếp từ vector t_i mà không cần phải thành lập ma trận S_i

$$\bar{u}_{jj} = \sqrt{\frac{q_{w_i} - \sum_{k=j}^{n} t_k^2}{q_{w_i} - \sum_{k=j+1}^{n} t_k^2}} \qquad j = 1:n \qquad (17a)$$

$$\bar{u}_{lj} = -\frac{t_lt_j}{\sqrt{\left(q_{w_i} - \sum_{k=j}^{n} t_k^2\right)\left(q_{w_i} - \sum_{k=j+1}^{n} t_k^2\right)}} \quad l = 1:j-1 \qquad (17b)$$

Nếu ký hiệu $\alpha_j = q_{w_i} - \sum_{k=j+1}^{n} t_k^2$ và $\alpha_j = q_{w_i} - \sum_{k=j}^{n} t_k^2 = \alpha_{j-1} - t_j^2$, thì thuật toán lập ma trận $\bar{U}$ sẽ đơn giản như sau:

$\alpha_0 = q_{w_i}$

for j = 1:n

$\alpha_j = \alpha_{j-1} - t_j^2$

$\bar{u}_{jj} = \sqrt{\frac{\alpha_j}{\alpha_{j-1}}}$

$\beta = -\frac{t_j}{\bar{u}_{jj}\alpha_{j-1}}$

for l = 1:j-1

$\bar{u}_{lj} = \beta.t_l$

end

end

Cần lưu ý rằng ma trận $\bar{U}$ là ma trận tam giác trên. Để tiết kiệm bộ nhớ và tăng tốc độ tính toán, ta nên lưu ma trận $\bar{U}$ thành dãy một chiều theo thứ tự các phần tử sau đây

$$\begin{pmatrix} 1 & 2 & 4 & 7... \\ & 3 & 5 & 8... \\ & & 6 & 9... \\ & & & 10... \end{pmatrix} \quad (18)$$

Đoạn chương trình MATLAB sau đây cho phép thành lập ma trận $\bar{U}$ một cách hiệu quả trên máy tính khi lưu trữ theo định dạng (18):

```
alpha0=qw;
for i=n:-1:1
    alpha1=alpha0-t(i)^2;
    U(i*(i+1)/2)=sqrt(alpha1/alpha0);
    beta=-  t(i)/U(i*(i+1)/2)/alpha0;
    for j=1:i-1
      U(j+i*(i-1)/2)=beta*t(j);
    end
    alpha0=alpha1;
end
```

Kích thước ma trận $\bar{U}$ bằng với ma trận U. Do đó việc duy trì ma trận $\bar{U}$ có thể làm cho bộ nhớ máy tính tăng lên gấp đôi. Quan sát việc nhân hai ma trận tam giác trên $U_{(i-1)}\bar{U}_i$ ta thấy kết quả nhân ma trận U với cột thứ n của ma trận $\bar{U}$ có thể lưu ngay vào cột thứ n của ma trận U. Vì cột này không tham gia vào việc tính n-1 cột đầu của ma trận U. Tương tự cột thứ n-1 không tham gia vào việc tính n-2 cột đầu, vv. Do đó nếu biến đổi ma trận U theo chiều từ phải sang trái, ta không cần thiết phải lập ma trận $\bar{U}$. Quá trình biến đổi này có thể thực hiện bằng đoạn chương trình MATLAB sau

$\alpha_0 = q_{w_i}$

for j = n:-1:1

$\alpha_j = \alpha_{j-1} - t_j^2$

$\beta = \sqrt{\frac{\alpha_j}{\alpha_{j-1}}}$

for l = 1:j

$u_{lj} = \beta.u_{lj}$

end

$\beta = -\frac{t_j}{\beta\alpha_{j-1}}$

for l = 1:j-1

$\gamma = \beta.t_l$

for k = 1:j

$u_{kj} = u_{kj} + \gamma.u_{kl}$

end

end

end

```
alpha0=qw;
for i=n:-1:1
  alpha1=alpha0-t(i)^2;
  U_ii=qrt(alpha1/alpha0);
  for j=1:i
    U(j+i*(i-1)/2)=U(j+i*(i-1)/2)*U_ii;
  end
  beta=-t(i)/U_ii/alpha0;
  for j=1:i-1
    U_ji=beta*t(j);
    for k=1:j
  U(k+i*(i-1)/2)=U(k+i*(i-
1)/2)+U(k+j*(j-1)/2)*U_ji;
  end
```

end

alpha0=alpha1;

end

Trong đoạn chương trình trên ma trận U được lưu trữ theo định dạng (18). Quy trình tính toán bình sai truy hồi theo thuật toán Carlson như sau

Bước 1: Bắt đầu với ma trận $U_{(0)} = 10^6 E$ giá trị ban đầu của vector ẩn số $\hat{X}_{(0)}$ và dạng toàn phương $\hat{\phi}_{(0)} = 0$

i = 1

Bước 2:

Tính vector t_i $t_i = U_{(i-1)}^T a_i^T$

Tính số hạng tự do của trị đo thứ i

$$w_i = f_i\left(\hat{X}_{(i-1)}\right) - y_i$$

Tính trọng số đảo của số hạng tự do l_i

$$q_{w_i} = \frac{1}{p_i} + t_i^T t_i$$

Kiểm tra $|w_i| < 3\sigma_0\sqrt{q_{w_i}}$

nếu không thỏa mãn chuyển sang bước 3

Ước lượng của ẩn số sau khi xử lý trị đo thứ i

$$\hat{X}_{(i)} = \hat{X}_{(i-1)} - U_{(i-1)} t_i \frac{w_i}{q_{w_i}}$$

Biến đổi ma trận $U_{(i-1)}$ thành $U_{(i)}$

Dạng toàn phương sau khi xử lý trị đo thứ i

$$\hat{\phi}_{(i)} = \hat{\phi}_{(i-1)} + \frac{w_i^2}{q_{w_i}}$$

Bước 3: i = i +1

Nếu i > n thì chuyển sang bước 4, ngược lại quay về bước 2

Bước 4: Xuất kết quả bình sai

Thuật toán Carlson có dạng tính toán tốt và thừa hưởng các đặc điểm về tính ổn định và độ chính xác của các bộ lọc căn bậc hai nói chung. Mặc dù công thức Carlson đòi hỏi ít khối lượng tính toán và bộ nhớ hơn phương pháp căn bậc hai của Potter, nó vẫn kém hiệu quả hơn thuật toán Kalman truyền thống. Vì không giống như Kalman truyền thống, thuật toán Carlson đòi hỏi n phép căn bậc hai mỗi khi cập nhật một trị đo mới, và phép lấy căn là một toán tử gần đúng và chiếm nhiều thời gian.

PGS.TSKH. Hà Minh Hòa trong một số tài liệu [5, 6, 7] đã đề nghị một thuật toán tương tự khi phân tích ma trận hiệp phương sai ở dạng $Q = T^{-1}T^{-T}$ và đặt tên là thuật toán T^{-T}. So sánh với (10), ta dễ dàng dẫn ra U = T^{-1}. Vì vậy T^{-1} cũng là ma trận tam giác trên. Điểm mới của thuật toán T^{-T} là quá trình biến đổi ma trận $T_{(i-1)}^{-T}$ thành $T_{(i)}^{-T}$ được thực hiện bằng phép biến đổi xoay Givens. Phương pháp này nổi tiếng về các đặc tính số ưu việt nên có thể giảm sai số làm tròn hơn nữa. Tuy nhiên thuật toán cũng có chung nhược điểm như trên là đòi hỏi n phép căn bậc hai mỗi khi cập nhật một trị đo mới.

2.2. Thuật toán phân tích ma trận hiệp phương sai U-D

Một cách tiếp cận mới đầy hứa hẹn cho bộ lọc Kalman liên quan đến việc phân tích ma trận phương sai thành tam giác mà không đòi hỏi các phép khai căn. Ma trận hiệp phương sai Q được phân tích thành

$$Q = U.D.U^T \qquad (19)$$

Trong đó U là ma trận tam giác trên đơn vị và D là ma trận chéo. Bierman [3] đã đề nghị sự phân tích này và dẫn ra một thuật toán cập nhật trị đo U-D. Dựa trên ý tưởng này chúng tôi sẽ chứng minh lại thuật toán cho

mục đích của mình. Do đó nó không hoàn toàn giống với thuật toán nguyên bản của Bierman [3]

Kết hợp (19) và (5), ta có

$$U_{(i)}D_{(i)}U_{(i)}^T = U_{(i-1)}S_iU_{(i-1)}^T \quad (20)$$

Trong đó

$$S_i = D_{(i-1)} - \frac{1}{q_{w_i}}t_it_i^T \quad (21)$$

$$t_i^T = a_iU_{(i-1)}D_{(i-1)} \quad (22)$$

Vì Q_i là ma trận đối xứng và xác định dương nên S_i cũng đối xứng và xác định dương. Do đó tồn tại duy nhất các ma trận tam giác trên đơn vị $\overline{U}$ và ma trận chéo $\overline{D}$ sao cho

$$S_i = \overline{U}_i\overline{D}_i\overline{U}_i^T \quad (23)$$

Thay (23) vào (20), ta được

$$U_{(i)}D_{(i)}U_{(i)}^T = U_{(i-1)}\overline{U}_i\overline{D}_i\overline{U}_i^TU_{(i-1)}^T \quad (24)$$

Suy ra $U_{(i)} = U_{(i-1)}\overline{U}_i$ (25)

và $D_{(i)} = \overline{D}_{(i)}$ (26)

Các phần tử của ma trận $\overline{D}_i$ và $\overline{U}_i$ có thể tính theo công thức sau (suy ra từ (17))

$$\overline{d}_j = d_j\frac{q_{w_i} - \sum_{k=j}^{n}\frac{t_k^2}{d_k}}{q_{w_i} - \sum_{k=j+1}^{n}\frac{t_k^2}{d_k}} \quad j = 1: n \quad (27a)$$

$$\overline{u}_{lj} = -\frac{t_lt_j}{d_j\left(q_{w_i} - \sum_{k=j}^{n}\frac{t_k^2}{d_k}\right)} \quad l = 1: j-1 \quad (27b)$$

Trong đó d_j là phần tử đường chéo thứ j của ma trận $D_{(i-1)}$

Tương tự nếu ký hiệu $\alpha_j = q_{w_i} - \sum_{k=j+1}^{n}\frac{t_k^2}{d_k}$ và $\alpha_j = q_{w_i} - \sum_{k=j}^{n}\frac{t_k^2}{d_k} = \alpha_{j-1} - \frac{t_j^2}{d_j}$, thì thuật toán lập ma trận $\overline{D}_i$ và $\overline{U}_i$ đơn giản như sau:

$\overline{U} = E$

$\alpha_0 = q_{w_i}$

for j = 1:n

$$\alpha_j = \alpha_{j-1} - \frac{t_j^2}{d_j}$$

$$\overline{d}_j = d_j\frac{\alpha_j}{\alpha_{j-1}}$$

$$\beta = -\frac{t_j}{d_j\alpha_j}$$

for l = 1:j-1

$\overline{u}_{lj} = \beta.t_l$

end

end

Đoạn chương trình MATLAB sau thể hiện cho thuật toán trên

```
U_=eye(n);
alpha0=qw;
for i=n:-1:1
    alpha1=alpha0-t(i)^2/D(i);
    D_(i)=D(i)*alpha1/alpha0;
    beta=-t(i)/D(i)/alpha1;
    for j=1:i-1
     U_(j+i*(i-1)/2)=beta*t(j);
    end
    alpha0=alpha1;
end
```

Tuy nhiên việc lưu trữ ma trận $\overline{U}$ và $\overline{D}$ là không có lợi, trong khi ta có thể biến đổi

trực tiếp $U_{(i)}$ và $D_{(i)}$ từ ma trận t_i và $D_{(i-1)}$. Đoạn chương trình MATLAB sau cho phép làm điều đó

$\alpha_0 = q_{w_i}$

for j = 1:-1:n

$\alpha_j = \alpha_{j-1} - \frac{t_j^2}{d_j}$

$d_j = d_j \frac{\alpha_j}{\alpha_{j-1}}$

$\beta = -\frac{t_j}{d_j \alpha_{j-1}}$

for l = 1:j-1

$\gamma = \beta . t_l$

for k = 1:j

$u_{kj} = u_{kj} + u_{kl} * \gamma$

end

end

end

```
alpha0=qw;
for i=n:-1:1
 alpha1=alpha0-t(i)^2/D(i);
 D(i)=D(i)*alpha1/alpha0;
 beta=-t(i)/D(i)/alpha0;
 for j=1:i-1
  U_ji=beta*t(j);
  for k=1:j
     U(k+i*(i-1)/2)=U(k+i*(i-
1)/2)+U(k+j*(j-1)/2)*U_ji;
  end
 end
 alpha0=alpha1;
end
```

Quy trình tính toán bình sai truy hồi theo thuật toán U-D như sau

Bước 1: Bắt đầu với ma trận $U_{(0)} = E$, $D_{(0)} = 10^6 E$, giá trị ban đầu của vector ẩn số $\hat{X}_{(0)}$ và dạng toàn phương $\hat{\phi}_{(0)} = 0$

i = 1

Bước 2:

Tính vector t_i $\quad t_i^T = a_i U_{(i-1)} D_{(i-1)}$

Tính số hạng tự do của trị đo thứ i

$$w_i = f_i\left(\hat{X}_{(i-1)}\right) - y_i$$

Tính trọng số đảo của số hạng tự do w_i

$$q_{w_i} = \frac{1}{p_i} + t_i^T D_{(i-1)}^{-1} t_i = \frac{1}{p_i} + \sum_{j=1}^{n} \frac{(t_i)_j^2}{d_j}$$

Kiểm tra $\quad |w_i| < 3\sigma_0 \sqrt{q_{w_i}}$

nếu không thỏa mãn chuyển sang bước 3

Ước lượng của ẩn số sau khi xử lý trị đo thứ i

$$\hat{X}_{(i)} = \hat{X}_{(i-1)} - U_{(i-1)} t_i \frac{w_i}{q_{w_i}}$$

Biến đổi ma trận $U_{(i-1)}$ thành $U_{(i)}$, $D_{(i-1)}$ thành $D_{(i)}$

Dạng toàn phương sau khi xử lý trị đo thứ i

$$\hat{\phi}_{(i)} = \hat{\phi}_{(i-1)} + \frac{w_i^2}{q_{w_i}}$$

Bước 3: $\quad$ i = i +1

Nếu i > n thì chuyển sang bước 4, ngược lại quay về bước 2

Bước 4: Xuất kết quả bình sai

Kaminsky vào năm 1971 (tham khảo trong tài liệu [3]) đã so sánh cẩn thận khối lượng tính toán của các thuật toán khi xử lý m trị đo cho n ẩn số. Một vài kết quả được cho ở bảng sau

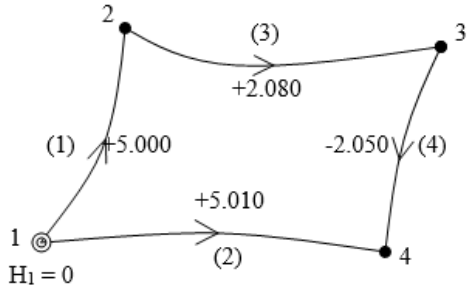
Bảng 1. So sánh khối lượng tính toán của các thuật toán

Thuật toán	Số phép cộng	Số phép nhân	Số phép chia	Số phép khai căn
Thuật toán Q	$(1.5n^2+3.5n)m$	$(1.5n^2+4.5n)m$	m	0
Carlson	$0.5n^2+0.5n+$ $(1.5n^2+3.5n)m$	$0.5n^2+0.5n+$ $(2.0n^2+5.0n)m$	2nm	nm
U-D	$0.5n^2-0.5n+$ $(1.5n^2+1.5n)m$	n^2-n+ $(1.5n^2+5.5n)m$	nm	0

Theo bảng trên ta thấy các thuật toán căn bậc hai đều có khối lượng tính toán lớn hơn thuật toán truyền thống. Trong số các thuật toán căn bậc hai thì thuật toán U-D có khối lượng tính toán ít nhất do không có các phép khai căn. Kaminsky cũng thông báo rằng các thuật toán căn bậc hai có thể cung cấp độ chính xác gấp đôi so với thuật toán Q trong những điều kiện xấu. Với các đặc tính số vượt trội và khối lượng tính toán hợp lý, cách tiếp cận căn bậc hai sẽ là lựa chọn hàng đầu trong các ứng dụng có cấu hình máy tính hạn chế hay bài toán xử lý trong điều kiện xấu.

3. Ví dụ minh họa

Để chứng minh tính đúng đắn của các thuật toán căn bậc hai, chúng tôi tính toán bình sai một mạng lưới thủy chuẩn đơn giản ở hình 1. Trong đó điểm 1 là điểm gốc, các chênh cao có trọng số đều bằng 1



Hình 1. Tuyến thủy chuẩn khép kín

3.1. Bình sai cổ điển

* Chọn vector ẩn số là $X=(H_2 \quad H_3 \quad H_4)^T$ với giá trị gần đúng là $X_0=(5.00 \quad 7.08 \quad 5.01)^T$

* Vector trị đo $Y=(+5.000 \quad +5.010 \quad +2.080 \quad -2.050)^T$

* Ma trận trọng số của trị đo P = E

* Ma trận hệ số $A=\begin{pmatrix}1&0&0\\0&0&1\\-1&1&0\\0&-1&1\end{pmatrix}$

* Vector số hạng tự do $W=A.X_0-Y=(0 \quad 0 \quad 0 \quad -0.02)^T$

* Ma trận phương trình chuẩn $N=A^TPA=\begin{pmatrix}2&-1&0\\&2&-1\\&&2\end{pmatrix}$

* Ma trận trọng số đảo của ẩn số $Q=N^{-1}=\begin{pmatrix}\frac{3}{4}&\frac{1}{2}&\frac{1}{4}\\&1&\frac{1}{2}\\&&\frac{3}{4}\end{pmatrix}$

* Vector số hạng tự do của phương trình chuẩn $b=A^TPW=(0 \quad -0.02 \quad +0.02)^T$

* Vector số hiệu chỉnh vào ẩn số $\Delta X = -Qb = \begin{pmatrix} +0.005 & +0.010 & -0.005 \end{pmatrix}^T$

* Vector ẩn số sau khi bình sai $\hat{X} = X_0 + \Delta X = \begin{pmatrix} 5.005 & 7.090 & 5.005 \end{pmatrix}^T$

* Vector phần dư của các trị đo $V = A.\Delta X - W = \begin{pmatrix} +0.005 & -0.005 & +0.005 & +0.005 \end{pmatrix}^T$

* Dạng toàn phương $\phi = V^T PV = 0.0001$

3.2. Bình sai truy hồi theo thuật toán Q

Xuất phát với $Q_{(0)} = \begin{pmatrix} 10^6 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 10^6 \end{pmatrix}$, $\hat{X}_{(0)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$ và $\hat{\phi}_{(0)} = 0$, ta lần lượt nhận được

* $w_1 = 0$, $a_1 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$, $q_{w_1} = 1 + 10^6$, $\hat{X}_{(1)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$, $Q_{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ & 10^6 & 0 \\ & & 10^6 \end{pmatrix}$, $\hat{\phi}_{(1)} = 0$

* $w_2 = 0$, $a_2 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$, $q_{w_2} = 1 + 10^6$, $\hat{X}_{(2)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$, $Q_{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ & 10^6 & 0 \\ & & 1 \end{pmatrix}$, $\hat{\phi}_{(2)} = 0$

* $w_3 = 0$, $a_3 = \begin{pmatrix} -1 & 1 & 0 \end{pmatrix}$, $q_{w_3} = 2 + 10^6$ $\hat{X}_{(3)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$, $Q_{(3)} = \begin{pmatrix} 1 & 1 & 0 \\ & 2 & 0 \\ & & 1 \end{pmatrix}$, $\hat{\phi}_{(3)} = 0$

* $w_4 = -0.02$, $a_4 = \begin{pmatrix} 0 & -1 & 1 \end{pmatrix}$, $q_{w_4} = 4$, $\hat{X}_{(4)} = \begin{pmatrix} 5.005 & 7.090 & 5.005 \end{pmatrix}^T$, $Q_{(4)} = \begin{pmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ & 1 & \frac{1}{2} \\ & & \frac{3}{4} \end{pmatrix}$, $\hat{\phi}_{(4)} = 0.0001$

3.3. Bình sai truy hồi theo thuật toán Carlson

Xuất phát với $U_{(0)} = \begin{pmatrix} 10^6 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 10^6 \end{pmatrix}$, $\hat{X}_{(0)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$ và $\hat{\phi}_{(0)} = 0$, ta lần lượt nhận được

* $t_1^T = \begin{pmatrix} 10^6 & 0 & 0 \end{pmatrix}$, $q_{w_1} = 1 + 10^{12}$, $\overline{U}_1 = \begin{pmatrix} 10^{-6} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{X}_{(1)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$, $U_{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 10^6 \end{pmatrix}$, $\hat{\phi}_{(1)} = 0$

* $t_2^T = \begin{pmatrix} 0 & 0 & 10^6 \end{pmatrix}$, $q_{w_2} = 1 + 10^{12}$, $\overline{U}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 10^6 \end{pmatrix}$, $\hat{X}_{(2)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$, $U_{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{\phi}_{(2)} = 0$

* $t_3^T = \begin{pmatrix} -1 & 10^6 & 0 \end{pmatrix}$, $q_{w_3} = 2 + 10^{12}$, $\overline{U}_3 = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & \sqrt{2}.10^{-6} & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{X}_{(3)} = \begin{pmatrix} 5.00 & 7.08 & 5.01 \end{pmatrix}^T$, $U_{(3)} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & \sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{\phi}_{(3)} = 0$

* $t_4^T = \begin{pmatrix} 0 & -\sqrt{2} & 1 \end{pmatrix}$, $q_{w_4} = 4$, $\overline{U}_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & \frac{\sqrt{6}}{6} \\ 0 & 0 & \frac{\sqrt{3}}{2} \end{pmatrix}$,

$\hat{X}_{(4)} = (5.005 \quad 7.090 \quad 5.005)^T$,

$U_{(4)} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{6}}{6} & \frac{\sqrt{3}}{6} \\ 0 & \frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} \\ 0 & 0 & \frac{\sqrt{3}}{2} \end{pmatrix}$, $\hat{\phi}_{(4)} = 0.0001$

3.3. Bình sai truy hồi theo thuật toán U-D

Xuất phát với $U_{(0)} = E$, $D_{(0)} = \begin{pmatrix} 10^6 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 10^6 \end{pmatrix}$,

$\hat{X}_{(0)} = (5.00 \quad 7.08 \quad 5.01)^T$ và $\hat{\phi}_{(0)} = 0$, ta lần lượt nhận được

* $t_1^T = (10^6 \quad 0 \quad 0)$, $q_{w_1} = 1 + 10^6$,

$\hat{X}_{(1)} = (5.00 \quad 7.08 \quad 5.01)^T$,

$D_{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 10^6 \end{pmatrix}$, $U_{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{\phi}_{(1)} = 0$

* $t_2^T = (0 \quad 0 \quad 10^6)$, $q_{w_2} = 1 + 10^6$,

$\hat{X}_{(2)} = (5.00 \quad 7.08 \quad 5.01)^T$, $D_{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & 1 \end{pmatrix}$,

$U_{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ $\hat{\phi}_{(2)} = 0$

* $t_3^T = (-1 \quad 10^6 \quad 0)$, $q_{w_3} = 10^6$,

$\hat{X}_{(3)} = (5.00 \quad 7.08 \quad 5.01)^T$, $D_{(3)} = \begin{pmatrix} \frac{1}{2} & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$,

$U_{(3)} = \begin{pmatrix} 1 & \frac{1}{2} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{\phi}_{(3)} = 0$

* $t_4^T = (0 \quad -2 \quad 1)$, $q_{w_4} = 4$,

$\hat{X}_{(4)} = (5.005 \quad 7.090 \quad 5.005)^T$,

$D_{(4)} = \begin{pmatrix} \frac{1}{2} & 0 & 0 \\ 0 & \frac{2}{3} & 0 \\ 0 & 0 & \frac{3}{4} \end{pmatrix}$, $U_{(4)} = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & 1 & \frac{2}{3} \\ 0 & 0 & 1 \end{pmatrix}$, $\hat{\phi}_{(4)} = 0.0001$

3.4. So sánh các kết quả bình sai

Các thuật toán khác nhau đều cho cùng kết quả trên vector ẩn số X và dạng toàn phương ϕ. Riêng ma trận hiệp phương sai của ẩn số, ta có

a) Thuật toán Carlson

$$U_{(4)}U_{(4)}^T = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{6}}{6} & \frac{\sqrt{3}}{6} \\ 0 & \frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} \\ 0 & 0 & \frac{\sqrt{3}}{2} \end{pmatrix} \begin{pmatrix} \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{6}}{6} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{3} & \frac{\sqrt{3}}{2} \end{pmatrix} = \begin{pmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ & 1 & \frac{1}{2} \\ & & \frac{3}{4} \end{pmatrix} = Q_{(4)} = Q$$

b) Thuật toán U-D

$$U_{(4)}D_{(4)}U_{(4)}^T = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & 1 & \frac{2}{3} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{2} & 0 & 0 \\ 0 & \frac{2}{3} & 0 \\ 0 & 0 & \frac{3}{4} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{3} & \frac{2}{3} & 1 \end{pmatrix} = \begin{pmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ & 1 & \frac{1}{2} \\ & & \frac{3}{4} \end{pmatrix} = Q$$

Vậy kết quả bình sai theo tất cả các thuật toán là hoàn toàn đồng nhất với nhau.

4. Kết luận

Chúng tôi đã tóm tắt các quy trình tính toán bình sai truy hồi truyền thống, thuật toán T^{-T} và các thuật toán căn bậc hai gồm Carlson và U-D.

Thuật toán Carlson tương tự như thuật toán T^{-T}, chỉ khác nhau ở các tính lặp ma trận U (hay T^{-1}). Chúng tôi đã dẫn dắt ra công thức truy hồi và chương trình tính lặp cho ma trận U. Tuy nhiên việc so sánh nó với phép biến đổi xoay Given trong thuật toán T^{-T} cần phải có những nghiên cứu thêm nữa.

So với thuật toán Carlson và T^{-T}, thuật toán U-D có ưu điểm là không phải thực hiện phép khai căn nào. Điều này có thể giúp thuật toán tốt và ổn định hơn về mặt tính toán số. Về mặt lập trình, thuật toán U-D tốn bộ nhớ hơn vì cần duy trì thêm 1 dãy để giữ các phần tử đường chéo của ma trận chéo D.

Chúng tôi dùng một mạng lưới thủy chuẩn khép kín để kiểm tra tính đúng đắn của các thuật toán. Các thuật toán đã nêu đều cho cùng kết quả bình sai.❍

Tài liệu tham khảo

[1]. Krakiwsky E.J., (1975), "A synthesis of recent advances in the method of least squares", Lecture Notes at University of New Brunswick, Fredericton, Canada.

[2]. Mikhail E.M and F. Ackermann, (1976), "Observation and Least Squares", Dun-Donnelley Publisher, New York, 497pp.

[3]. Bierman G.J., (1977), "Factorization methods for discrete sequential estimation", Academic Press, 241pp.

[4]. Markuze U.I, (1989), "Các thuật toán bình sai lưới trắc địa trên máy tính", Nhà xuất bản Nedra, 247pp.

[5]. Hà Minh Hòa, (2002), "Nghiên cứu một thuật toán bình sai mạng lưới trắc địa tự do", Tạp chí Địa chính, 10-2002, 9-12.

[6]. Hà Minh Hòa và Nguyễn Ngọc Lâu, (2005), "Những phát triển trong việc xử lý dữ liệu GPS để nghiên cứu chuyển dịch của vỏ trái đất tại Việt Nam", Hội nghị Khoa học và Công nghệ lần thứ 9 tại Đại học Bách Khoa TP Hồ Chí Minh.

[7]. Hà Minh Hòa và Bùi Đăng Quang, (2009), "Phát triển thuật toán triển khai mô hình bình sai tổng quát đối với các mạng lưới trắc địa", Tạp chí Khoa học Đo đạc và Bản đồ, số 2, 12-2009, 13-20.

[8]. Nguyễn Ngọc Lâu, Nguyễn Thị Thanh Hương, (2013) Xác định các mô hình sai số cho trị đo GPS và GLONASS. Tạp chí Khoa học Đo đạc và Bản đồ số 18, tháng 12 năm 2013, 11-18.

[9]. Nguyễn Thị Thanh Hương, (2014) Phương pháp tìm kiếm các trị đo thô trong quá trình tính toán bình sai truy hồi với phép biến đổi xoay đối với mạng lưới độ cao hạng I, II quốc gia. Tạp chí Khoa học Đo đạc và Bản đồ số 22, tháng 12 năm 2014, 11-16.❍

Summary

Application of square root algorithms to sequentially adjust geodetic networks

Nguyen Ngoc Lau, Ho Chi Minh City University of Technology, Vietnam

Nguyen Thi Thanh Huong, Institute of Geodesy and Cartography, Vietnam

To adjust the geodetic networks effectively by computer, people often apply the algorithm of recursive/sequential adjustment. Of which the most commonly used is the Q algorithm. The Q algorithm maintains iterative computation on the symmetric variance-covariance matrix, which is considered to be sometimes difficult in numerical computations and reduces the accuracy of adjusted results. This paper has studied square root algorithms with better numerical stability and applied them to geodetic network adjustment. Based on the derived algorithms, we have designed MATLAB programs to use the least amount of computer memory, and use an illustrated example of a leveling network to demonstrate their correctness.❍