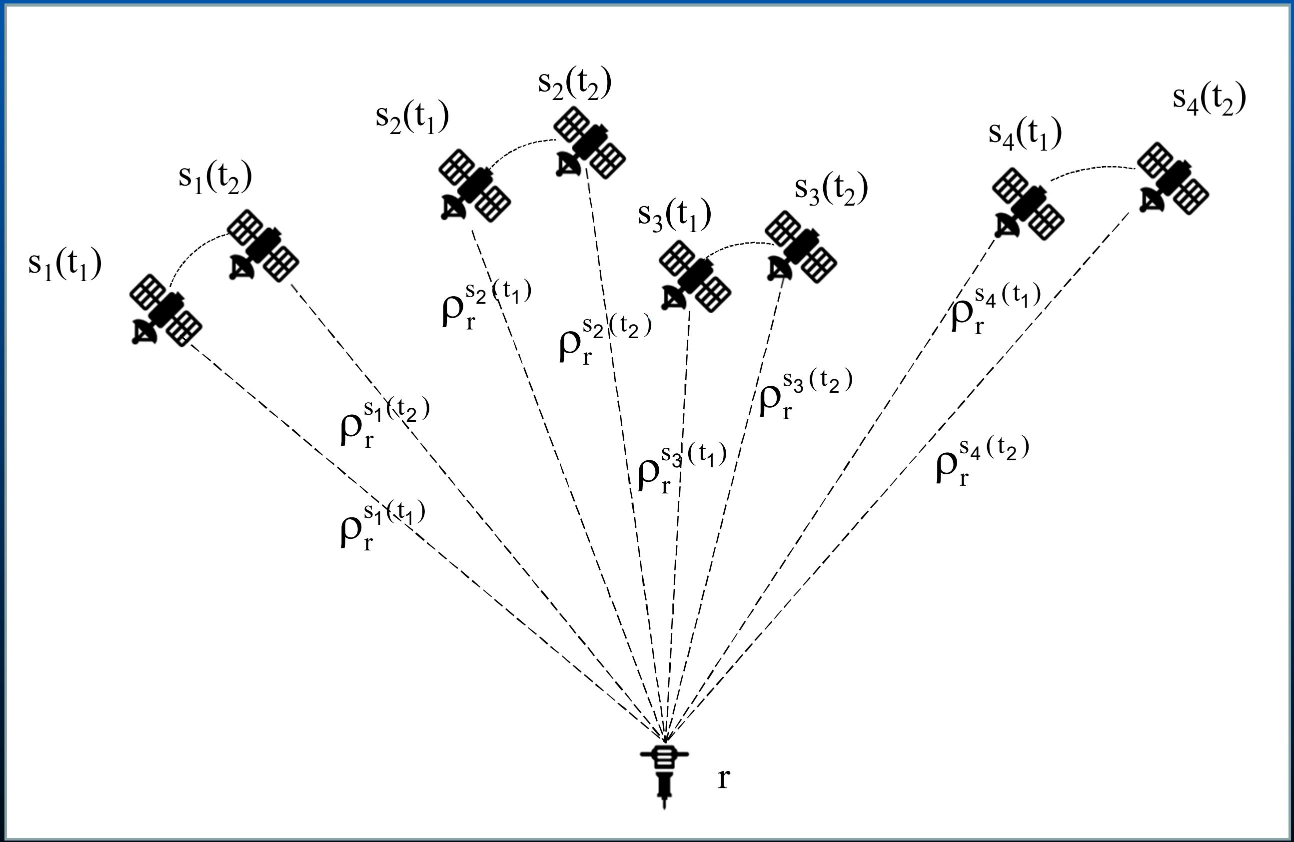
Multi-epoch SPP

k-epochs; n-satellites

number of observations: k·n
number of parameters: 3(position coordinates) + k (receiver clock bias)

redundancy: k·n –k – 3



Matrices of the residual equations in single epoch

The structure of normal equation matrices in j-th epoch

$$V_j = \begin{bmatrix} v^{s1} \\ v^{s2} \\ \vdots \\ v^{sn} \end{bmatrix}$$

$$A_j = \begin{bmatrix} \frac{x_r - x^{s1}}{\rho_r^{0s1}} & \frac{y_r - y^{s1}}{\rho_r^{0s1}} & \frac{z_r - z^{s1}}{\rho_r^{0s1}} & 1 \\ \frac{x_r - x^{s2}}{\rho_r^{0s2}} & \frac{y_r - y^{s2}}{\rho_r^{0s2}} & \frac{z_r - z^{s2}}{\rho_r^{0s2}} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{x_r - x^{sn}}{\rho_r^{0sn}} & \frac{y_r - y^{sn}}{\rho_r^{0sn}} & \frac{z_r - z^{sn}}{\rho_r^{0sn}} & 1 \end{bmatrix} = \begin{bmatrix} \underset{nx3}{A_{xj}} & \underset{nx1}{\mathbf{1}} \end{bmatrix}$$

$$X = \begin{bmatrix} dx_r \\ dy_r \\ dz_r \\ cdt_{rj} \end{bmatrix}$$

$$P_j = \begin{bmatrix} m_{P_r^{s1}}^{-2} & & & \\ & m_{P_r^{s2}}^{-2} & & \\ & & \ddots & \\ & & & m_{P_r^{sn}}^{-2} \end{bmatrix}$$

$$L_j = \begin{bmatrix} \left(P_r^{s1} - dT_r^{s1} - dI_r^{s1} - \rho_r^{0s1}\right) \\ \left(P_r^{s2} - dT_r^{s2} - dI_r^{s2} - \rho_r^{0s2}\right) \\ \vdots \\ \left(P_r^{sn} - dT_r^{sn} - dI_r^{sn} - \rho_r^{0sn}\right) \end{bmatrix}$$

Multi-epoch case

number of epochs: k

$$A = \begin{bmatrix} \underset{n\times 3}{A_{x1}} & \underset{n\times 1}{\mathbf{1}} & & \\ \underset{n\times 3}{A_{x2}} & & \underset{n\times 1}{\mathbf{1}} & \\ \vdots & & & \ddots & \\ \underset{n\times 3}{A_{xk}} & & & & \underset{n\times 1}{\mathbf{1}} \end{bmatrix}$$

$$X = \begin{bmatrix} dx_r \\ dy_r \\ dz_r \\ cdt_{r1} \\ \vdots \\ cdt_{rk} \end{bmatrix}$$

$$P = \begin{bmatrix} P_1 & & & \\ & P_2 & & \\ & & \ddots & \\ & & & P_k \end{bmatrix}$$

$$L = \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_k \end{bmatrix}$$

$$A^T PAX - A^T PL = 0 \longrightarrow X = (A^T PA)^{-1} A^T PL$$

Accuracy analysis

$A^TPAX - A^TPL = 0 \longrightarrow X = (A^TPA)^{-1}A^TPL$

Parameter covariance matrix

$$m_0^2 = \frac{V^TPV}{n-m}$$

$$C_x = m_0^2\left(A^TPA\right)^{-1}$$

Structure of the C_x matrix:

$$C_x = \begin{bmatrix} C_{rp} & C_{12} \\ C_{21} & C_{dt} \end{bmatrix}$$

$$C_{rp} = \begin{bmatrix} m_x^2 & m_{xy} & m_{xz} \\ m_{yx} & m_y^2 & m_{yz} \\ m_{zx} & m_{zy} & m_z^2 \end{bmatrix}$$

$$C_{dt} = \begin{bmatrix} m_{dt1}^2 & \cdots & m_{dt1k} \\ \vdots & \ddots & \vdots \\ m_{dtk1} & \cdots & m_{dtk}^2 \end{bmatrix}$$

squared mean errors of position coordinates

Ex 4 Single Point Positioning multi-epoch case

Data:

x^s ,y^s, z^s – satellite coordinates

x_r, y_r, z_r – approximate point position

P^{si} – code observation set

m = 30 cm – mean error of code observation (for forming weight matrix)

Task:

Prepare Matlab function for forming matrices: A, L, P and then conducting least squares adjustment process

Results:

X – increments to coordinates of approximate point position

C_x – covariance matrix of the X vector

m_0^2 – variance factor

m_i – mean errors of coordinates

Main function of Ex4

```
[dxi,v,vtpv,qx,m0,cx] = Ex4_SPP_multiepoch_studentname(prov,xyzsat_rov, xyzapp, nsat, nepoch)

clc
[nrsat, xref, yref, zref, pref, L1ref,L2ref, xrov,yrov, zrov, prov, L1rov, L2rov]...
    = textread('filename','%d%f%f%f%f%f%f%f%f%f%f%f%f','headerlines',6);
nsat = 7;
nepoch = 5;
xyzapp = [...; ...;  ...];
xyzsat_rov=[xrov,yrov, zrov];
.
.
.%creating matrices: A, P, L
.%adjustment
.%accuracy analysis
.
.
```

Some useful Matlab functions

[nr,x,y, …] = textread('filename','%d%f%f…' ,'headerlines',c) - read data from text file
nr is integer vector; x, y, … are real number vectors
c is the number of lines from the top of the file, which have to be omitted

Matrix operations:

a+b; a-b; a*b – addition, subtraction, multiplication
a = [1 2 3;4 5 6;7 8 9] b=a(:,1) = [1;4;7] c=a(2,:)=[2 5 8] d=a(2:3,2) = [5;8]
e = d'= [5 8] (transpose)
h = inv(g) (inverse)
n = norm(d) – 2-norm of the d vector (useful for calculating distance from coordinates)
f = diag(p) – get diagonal elements or create diagonal matrix
g = blkdiag(p1,p2) - create block diagonal matrix