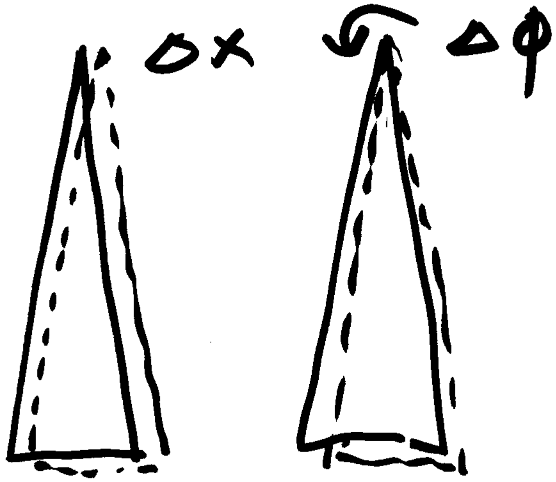
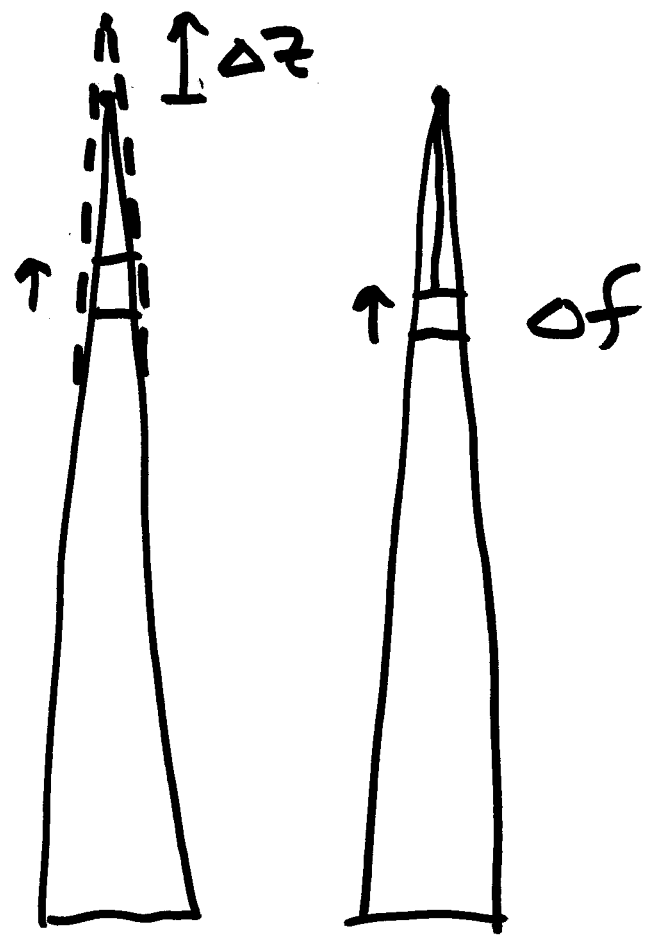
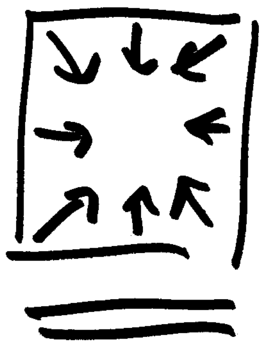


, Δy, Δw

small F.O.V.

Δz, Δf





monitor numerical stability

condition number $\approx \dfrac{\lambda_{max}}{\lambda_{min}}$

matrix N, cond(N)

$> \sim 10^{16}$ matlab error message

$> \sim 10^{12}$ unstable solution

$< \sim 10^{8}$ prefer

parameter selection problem

- small misclosure
- few parameters
- small condition number

See annotations along with code, made a few revision to actually make it work! recommend also putting semicolons after most statement to prevent their cluttering up the screen with unneeded numeric output

qbresect.m

```
% model based on polynomial corrections
% to ephemeris data
NPAR = 18
par = zeros(18,1)
% dx0, dx1, dx2, dy0, dy1, dy2, dz0, dz1, dz2
% dw0, dw1, dw2, dp0, dp1, dp2, dk0, dk1, dk2

puse [1; 4; 7] ;
```

puse = [1;4;7];

```
[m,n] = size(puse)

npuse = m
```

```
% parameter index array
use = zeros(NPAR, 1)
for i = 1: NPAR
    for j = 1: npuse
        if(puse(j) == i)
            use(i) = 1
            end;
        end;
    end;
% read support data
eph = dlmread('eph-cdf.txt');
att = dlmread('att-cdf.txt');
```

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

```
% read GCPs
[id,pd,pm,ps,ld,lm,ls,h,l,s] =
   textread('gcp.txt','%s %f %f %f
      %f %f %f %f %f %f');
[m,n] = size(pd);
npts = m;
% init NL LS arrays
N = zeros(npuse,npuse);
t = zeros(npuse,1);
B = zeros(2*npts,npuse);
```

```
f = zeros(2*npts,1);
delta = zeros(NPAR,1);
for i=1:NPAR
    delta(i)= 1.0e-06 ;
    end
% change values based on units
% LS iteration loop
for iter = 1:10     % hardwire @ 10
    nxeqn = 1
    nyeqn = 2
    dispv = zeros(ntps,3)     #, Vx, Vy
```

I found this worked better if position deltas and angle deltas were different:

```
for i=1:9
  delta(i)=1.0e-03
  end

for i=10:18
  delta(i)=1.0e-06
  end
```

```
rmsl = 0
rmss = 0
% build cond. equ.
for j = 1 : npts
    % convert GCP's
    phi = (pd(j) + pm(j)/60 + ps(j)/3600)
          * (pi/180) ;
    lam = (ld(j) + lm(j)/60 + ls(j)/3600)
          * (pi/180) ;
    ht = h(j) ;
    line = l(j)
    sample = s(j)
```

lam = - (ld(j) + lm(j)/60 ......

remember longitude is negative west of greenwich !!

```
% nominal value of cond. eqn.
F = QB(line, sample, phi, lam, ht, eph,
       att, par) ;          % { l - comp (l)
                            % { s - comp (s)

% V + B0 = f
dispv(j,1) = j ;
dispv(j,2) = F(1) ;
dispv(j,3) = F(2) ;
rmsl = rmsl + F(1) ^2 ;
rmss = rmss + F(2) ^2 ;
col = 1 ;
```

```
for i = 1:NPAR
  if (use(i) == 1)
    % compute ∂Fx/∂p , ∂Fy/∂p numerically
    % + fill coeff mx B, f
    pardel = par ;
    pardel(i) = pardel(i) + delta(i)
    Fdel = QB(line, sample, phi, lam,
      ht, eph, att, pardel);
```

% $\frac{\partial F}{\partial x} = \lim_{\Delta x \to 0} \frac{F(x+\Delta x) - F(x)}{\Delta x}$

% $\frac{\partial F}{\partial x} \approx \frac{F(x+\Delta x) - F(x)}{\Delta x}$

```
dFxdp = (Fdel(1) - F(1)) / delta(i) ;
dFydp = (Fdel(2) - F(2)) / delta(i) ;
 B(nxeqn, col) = dFxdp ;
 B(nyeqn, col) = dFydp ;
 col = col + 1
 f(nxeqn) = - F(1) ;
 f(nyeqn) = - F(2) ;
 end % use(i) == 1
end % parameter loop
nxeqn = nxeqn + 2 ;
nyeqn = nyeqn + 2 ;
end % end point loop
```

```
N = B' * B ;        % B^T W B      σ_0^2 = σ^2
t = B' * f ;                       σ = 2 pixels
cond_num = cond(N)

del = inv(N) * t ;
disp('parameter corrections');
del
% apply corrections to parameters
col=1
for i=1 : NPAR
    if(use(i) == 1)
        par(i) = par(i) + del(col)
        col = col + 1
    end
end
```

we are assuming W = I

```
rmsl = sqrt (rmsl/npts) ;
rmss = sqrt (rmss/npts) ;
 % next iteration
end   % for iter = 1:10
% look @ mag of del
% look @ stability of v^T v
```

terminate iterations

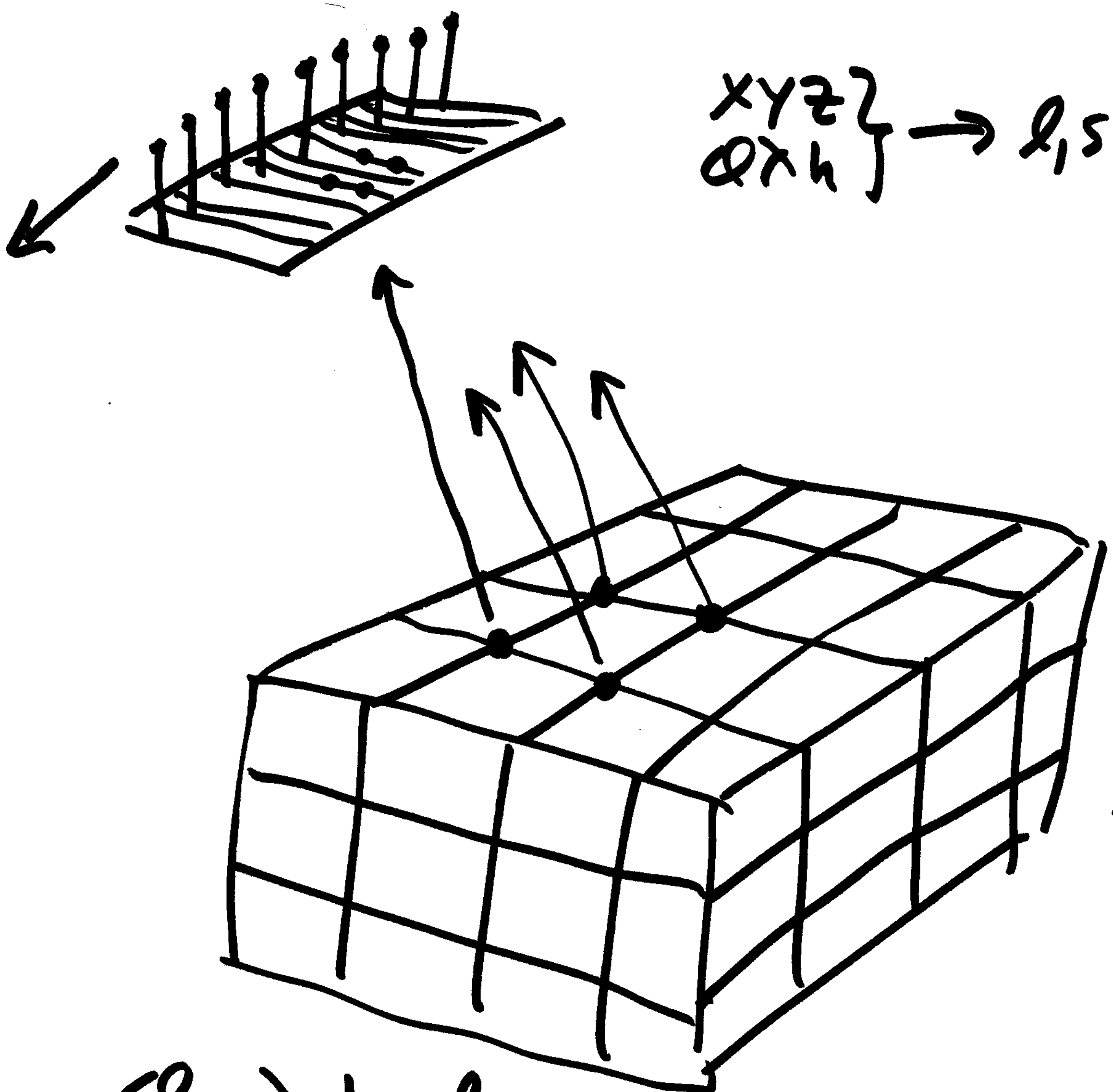
test convergence

```
disp('residuals') ;
dispV             % #, Vx, Vy ~ 1-2 pixel
disp ('rms l + s')
 rmsl
 rmss
```

```
disp('final parameters') ;
par
```

---

with refined parameters



$\left.\begin{matrix} xyz \\ \alpha\lambda h \end{matrix}\right\} \rightarrow l, s$

not closed form

$\left.\begin{matrix} 2\ \text{Eq} \\ 2\ \text{unk} \end{matrix}\right\}$ newton iter

$$\overbrace{\begin{matrix} X_1 & Y_1 & Z_1 \\ X_2 & Y_2 & Z_2 \\ X_3 & Y_3 & Z_3 \\ & \vdots & \\ X_{1000} & Y_{1000} & Z_{1000} \end{matrix}} \quad \overbrace{\begin{matrix} l_1 & s_1 \\ l_2 & s_2 \\ l_3 & s_3 \\ \vdots & \\ l_{1000} & s_{1000} \end{matrix}}$$

$\alpha \quad \lambda \quad h \quad l \quad s$

$\vdots$

$$l = \frac{a_0 + a_1 X + a_2 Y + \cdots}{1 + b_1 X + b_2 Y + \cdots} \quad (\to 3^{rd} \text{ order})$$

$$s = \frac{c_0 + c_1 X + c_2 Y + \cdots}{1 + d_1 X + d_2 Y + \cdots}$$

$1, X, Y, Z, XY, XZ, YZ, X^2, Y^2, Z^2, X^2Y, X^2Z, Y^2X,$
$Y^2Z, Z^2X, Z^2Y, XYZ, X^3, Y^3, Z^3$

20

$a_0, a_1 \ldots \quad b_0, b_1, \ldots$

RPC

80 unknowns
−2
78 unknowns