Chapter 5 Project: Apply Nelder-Mead to the Rheology Problem

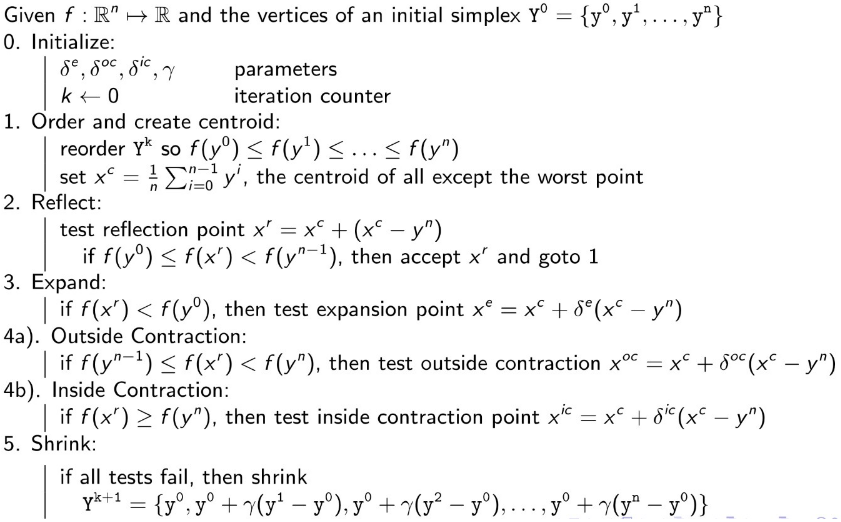
Matthew Saurette, Tyler Weames, and Sarah Wyse

Math 462
University of British Columbia - Okanagan

December 2020

The Rheology Problem

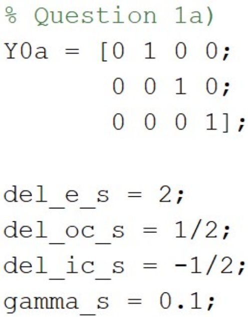
Nelder-Mead Algorithm



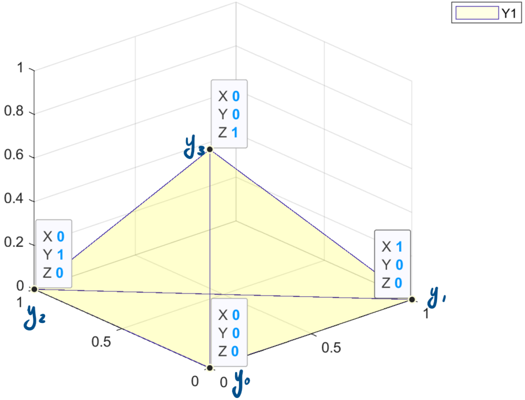
```
Given $f : \mathbb{R}^n \mapsto \mathbb{R}$ and the vertices of an initial simplex $\mathrm{Y}^0 = \{\mathrm{y}^0, \mathrm{y}^1, \ldots, \mathrm{y}^n\}$
0. Initialize:
    $\delta^e, \delta^{oc}, \delta^{ic}, \gamma$    parameters
    $k \leftarrow 0$    iteration counter
1. Order and create centroid:
    reorder $\mathrm{Y}^k$ so $f(y^0) \leq f(y^1) \leq \ldots \leq f(y^n)$
    set $x^c = \frac{1}{n} \sum_{i=0}^{n-1} y^i$, the centroid of all except the worst point
2. Reflect:
    test reflection point $x^r = x^c + (x^c - y^n)$
      if $f(y^0) \leq f(x^r) < f(y^{n-1})$, then accept $x^r$ and goto 1
3. Expand:
    if $f(x^r) < f(y^0)$, then test expansion point $x^e = x^c + \delta^e(x^c - y^n)$
4a). Outside Contraction:
    if $f(y^{n-1}) \leq f(x^r) < f(y^n)$, then test outside contraction $x^{oc} = x^c + \delta^{oc}(x^c - y^n)$
4b). Inside Contraction:
    if $f(x^r) \geq f(y^n)$, then test inside contraction point $x^{ic} = x^c + \delta^{ic}(x^c - y^n)$
5. Shrink:
    if all tests fail, then shrink
      $\mathrm{Y}^{k+1} = \{\mathrm{y}^0, \mathrm{y}^0 + \gamma(\mathrm{y}^1 - \mathrm{y}^0), \mathrm{y}^0 + \gamma(\mathrm{y}^2 - \mathrm{y}^0), \ldots, \mathrm{y}^0 + \gamma(\mathrm{y}^n - \mathrm{y}^0)\}$
```

Algorithm from lecture slides

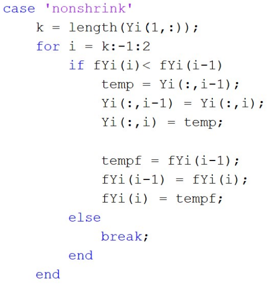
0. Initialize



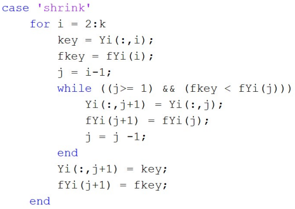
```
% Question 1a)
Y0a = [0 1 0 0;
       0 0 1 0;
       0 0 0 1];

del_e_s = 2;
del_oc_s = 1/2;
del_ic_s = -1/2;
gamma_s = 0.1;
```



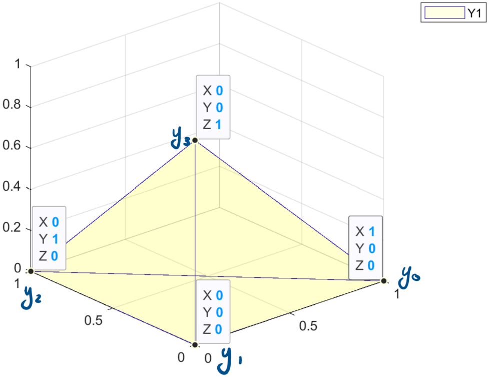
1. Order



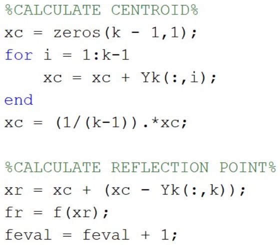
```
case 'nonshrink'
    k = length(Yi(1,:));
    for i = k:-1:2
        if fYi(i)< fYi(i-1)
            temp = Yi(:,i-1);
            Yi(:,i-1) = Yi(:,i);
            Yi(:,i) = temp;

            tempf = fYi(i-1);
            fYi(i-1) = fYi(i);
            fYi(i) = tempf;
        else
            break;
        end
    end
```



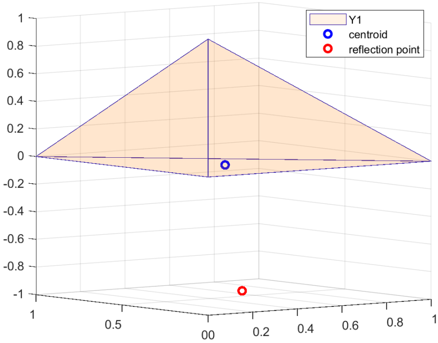
```
case 'shrink'
    for i = 2:k
        key = Yi(:,i);
        fkey = fYi(i);
        j = i-1;
        while ((j>= 1) && (fkey < fYi(j)))
            Yi(:,j+1) = Yi(:,j);
            fYi(j+1) = fYi(j);
            j = j -1;
        end
        Yi(:,j+1) = key;
        fYi(j+1) = fkey;
    end
```



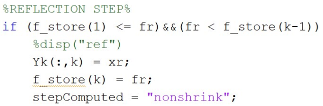
1 and 2. Calculate centroid and x^r



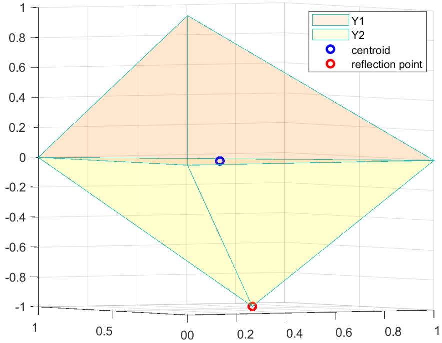
```
%CALCULATE CENTROID%
xc = zeros(k - 1,1);
for i = 1:k-1
    xc = xc + Yk(:,i);
end
xc = (1/(k-1)).*xc;

%CALCULATE REFLECTION POINT%
xr = xc + (xc - Yk(:,k));
fr = f(xr);
feval = feval + 1;
```



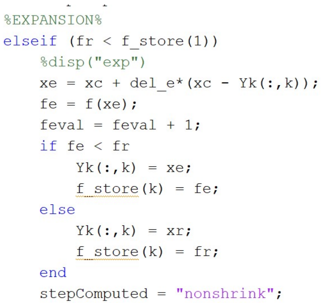
2. Reflect



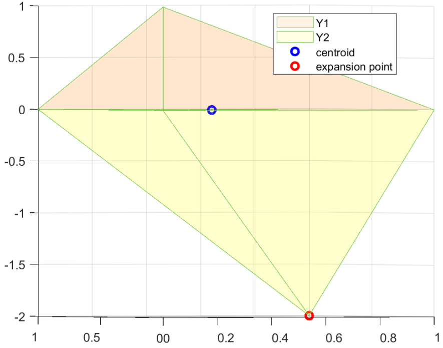
```
%REFLECTION STEP%
if (f_store(1) <= fr)&&(fr < f_store(k-1))
    %disp("ref")
    Yk(:,k) = xr;
    f_store(k) = fr;
    stepComputed = "nonshrink";
```



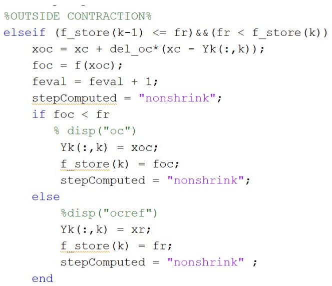
3. Expand



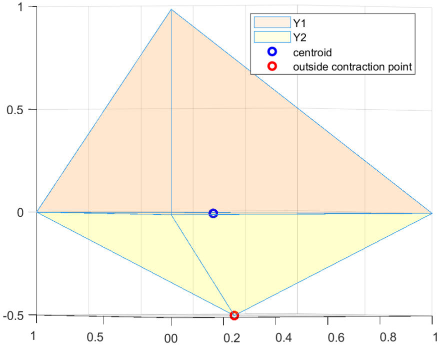
```
%EXPANSION%
elseif (fr < f_store(1))
    %disp("exp")
    xe = xc + del_e*(xc - Yk(:,k));
    fe = f(xe);
    feval = feval + 1;
    if fe < fr
        Yk(:,k) = xe;
        f_store(k) = fe;
    else
        Yk(:,k) = xr;
        f_store(k) = fr;
    end
    stepComputed = "nonshrink";
```



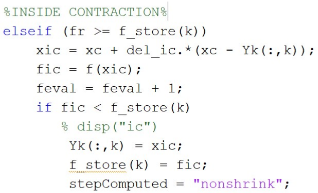
4.a) Outside Contraction



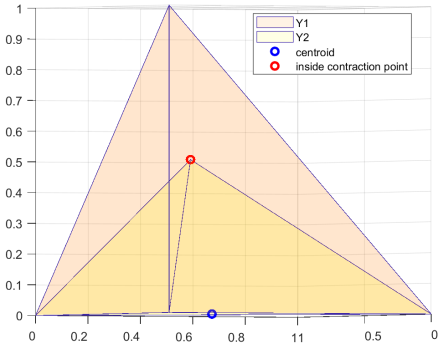
```
%OUTSIDE CONTRACTION%
elseif (f_store(k-1) <= fr)&&(fr < f_store(k))
    xoc = xc + del_oc*(xc - Yk(:,k));
    foc = f(xoc);
    feval = feval + 1;
    stepComputed = "nonshrink";
    if foc < fr
        % disp("oc")
        Yk(:,k) = xoc;
        f_store(k) = foc;
        stepComputed = "nonshrink";
    else
        %disp("ocref")
        Yk(:,k) = xr;
        f_store(k) = fr;
        stepComputed = "nonshrink" ;
    end
```



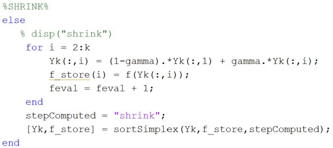
4.b) Inside Contraction



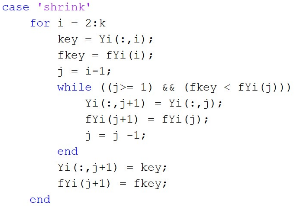
```
%INSIDE CONTRACTION%
elseif (fr >= f_store(k))
    xic = xc + del_ic.*(xc - Yk(:,k));
    fic = f(xic);
    feval = feval + 1;
    if fic < f_store(k)
       % disp("ic")
        Yk(:,k) = xic;
        f_store(k) = fic;
        stepComputed = "nonshrink";
```



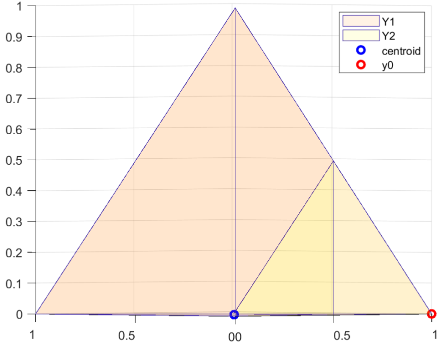
5. Shrink



```
%SHRINK%
else
   % disp("shrink")
    for i = 2:k
        Yk(:,i) = (1-gamma).*Yk(:,1) + gamma.*Yk(:,i);
        f_store(i) = f(Yk(:,i));
        feval = feval + 1;
    end
    stepComputed = "shrink";
    [Yk,f_store] = sortSimplex(Yk,f_store,stepComputed);
end
```



```
case 'shrink'
    for i = 2:k
        key = Yi(:,i);
        fkey = fYi(i);
        j = i-1;
        while ((j>= 1) && (fkey < fYi(j)))
            Yi(:,j+1) = Yi(:,j);
            fYi(j+1) = fYi(j);
            j = j -1;
        end
        Yi(:,j+1) = key;
        fYi(j+1) = fkey;
    end
```


Y1
Y2
centroid
y0