Homework 5
Shengchao Liu

1. Let $\bigtriangledown f_k = r_k = Ax_k - b$

   First prove that $r_k^T p_j = 0$, for $j = 0, 1, \dots k-1$

   Prove this by deduction:

   According to definition of α_k, we can easily get $r_{k+1}^T p_k = r_k^T p_k + \alpha_k p_k^T A p_k = 0$

   Thus $r_1^T p_0 = 0$ and equation stands for all $j = k-1$

   Suppose that inductive proof $r_k^T p_j = 0$ stands for k and $\forall j < k$, then we need to prove this also stands for $k+1$ and $\forall j < k-1$.

   Then $r_{k+1} = r_k + \alpha_k A p_k$

   So $p_j^T r_{k+1} = p_j^T r_k + \alpha_k p_j^T A p_k$, where $j < k-1$.

   And $p_j^T A p_k = 0$, $p_j^T r_k = 0$, $\forall j < k-1$, so $p_j^T r_{k+1} = 0$ and $\forall j < k-1$.

   To sum up, we can get $r_k^T p_j = 0$, for $j = 0, 1, \dots k-1$.

   And according to $r_k = -p_k + \beta_k p_{k-1}$

   We can get that $r_{k+1}^T r_k = -r_{k+1}^T p_k + \beta_{k+1} r_{k+1}^T p_{k-1} = 0$.

   So that $r_{k+1}^T r_k = 0$

   PR:

   $\beta_{k+1} = \frac{\bigtriangledown f_{k+1}^T (\bigtriangledown f_{k+1} - \bigtriangledown f_k)}{\|\bigtriangledown f_k\|^2} = \frac{\|f_{k+1}\|^2}{\|\bigtriangledown f_k\|^2}$

   HS:

   with exact line search, we have $\bigtriangledown f_k^T \cdot p_{k-1} = 0$

   $\beta = \frac{\bigtriangledown f_{k+1}^T (\bigtriangledown f_{k+1} - \bigtriangledown f_k)}{(\bigtriangledown f_{k+1} - \bigtriangledown f_k)^T p_k} = \frac{\bigtriangledown f_{k+1}^T \bigtriangledown f_{k+1}}{-\bigtriangledown f_k^T p_k}$

   And $p_k = -\bigtriangledown f_k + \beta_{k+1} p_{k-1}$, so $-\bigtriangledown f_k^T p_k = \bigtriangledown f_k^T \bigtriangledown f_k$

   So we can get $\beta = \frac{\bigtriangledown f_{k+1}^T \bigtriangledown f_{k+1}}{\bigtriangledown f_k^T \bigtriangledown f_k}$

2. According to Sherman-Morrison formula, we get:

   $\bar{A}^{-1} = A^{-1} - \frac{A^{-1} a b^T A^{-1}}{1 + b^T A^{-1} a}$

   Because we have $B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k} = B_k + \frac{y_k - B_k s_k}{\delta} \cdot \frac{(y_k - B_k s_k)^T}{\delta}$,

   where $\delta = \sqrt{(y_k - B_k s_k)^T s_k}$

   And put $A = B_k$, $\bar{A} = H_k$, $a = b = \frac{y_k - B_k s_k}{\delta}$ into the formula, we get:

   $H_{k+1} = H_k - \frac{H_k a b^T H_k}{1 + b^T H_k a} = H_k - \frac{H_k (y_k - B_k s_k)(y_k - B_k s_k)^T H_k}{(y_k - B_k s_k)^T s_k + (y_k - B_k s_k)^T H_k (y_k - B_k s_k)}$

   $(y_k - B_k s_k)^T H_k = (H_k (y_k - B_k s_k))^T = (H_k y_k - s_k)^T$

   $(y_k - B_k s_k)^T s_k + (y_k - B_k s_k)^T H_k (y_k - B_k s_k) = (y_k - B_k s_k)^T H_k B_k s_k + (y_k - B_k s_k)^T H_k (y_k - B_k s_k) = (y_k - B_k s_k)^T H_k (B_k s_k + y_k - B_k s_k) = (y_k - B_k s_k)^T H_k y_k = (H_k y_k - s_k)^T y_k$

So we can finally get $H_{k+1} = H_k - \frac{H_k(y_k - B_k s_k)(y_k - B_k s_k)^T H_k}{(y_k - B_k s_k)^T s_k + (y_k - B_k s_k)^T H_k (y_k - B_k s_k)} = H_k - \frac{(H_k y_k - s_k)(H_k y_k - s_k)^T}{(H_k y_k - s_k)^T y_k}$
$= H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}$

3. First is my answer:

   As I tested, the FR is very sensitive to parameters. If we choose a very good parameter set, it can perform as well as CG PR+, but if they are badly chosen, it took much longer time, over 200 epoches to get converged in this problem. And CG PR+ is the most stable one as I tested, the performance is almost the same, and it can converge very fast, within 60 epoches. And SGD is the slowest, in my testing cases, it will not get converged within 10000 epoches.

   Codes are below:

   CG_RF.m

```
function [inform, x] = CG_FR(fun, x, nonCGparams)

stepSizeParam = struct('c1',0.01, 'c2', 0.3, 'maxit',100);
x.f = feval(fun, x.p, 1);
x.g = feval(fun, x.p, 2);
p = -x.g;

for iter = 1 : nonCGparams.maxit
    [alpha, x_neo] = StepSize(fun, x, p, 1, stepSizeParam);
    beta = x_neo.g' * x_neo.g / (x.g' * x.g);
    p = -x_neo.g + beta * p;
    x = x_neo;
    if norm(x.g, Inf) <= nonCGparams.toler * (1+abs(x.f))
        inform.status = 1;
        inform.iter = iter;
        return;
    end
end

inform.status = 0;
inform.iter = nonCGparams.maxit;
return;
```

   CG_PRplus.m

```
function [inform, x] = CG_PRplus(fun, x, nonCGparams)

stepSizeParam = struct('c1',0.01, 'c2', 0.3, 'maxit',100);
x.f = feval(fun, x.p, 1);
```

```
x.g = feval(fun, x.p, 2);
p = -x.g;

for iter = 1 : nonCGparams.maxit
    [alpha, x_neo] = StepSize(fun, x, p, 1, stepSizeParam);
    beta = max(0, x_neo.g' * (x_neo.g - x.g) / norm(x.g,2)^2);
    p = -x_neo.g + beta * p;
    x = x_neo;
    if norm(x.g, Inf) <= nonCGparams.toler * (1+abs(x.f) )
        inform.status = 1;
        inform.iter = iter;
        return;
    end
end

inform.status = 0;
inform.iter = nonCGparams.maxit;
return;
```

SteepDescent.m

```
function [inform, x] = SteepDescent(fun, x, sdparams)

stepSizeParam = struct('c1',0.01, 'c2', 0.3, 'maxit',100);
x.f = feval(fun, x.p, 1);
x.g = feval(fun, x.p, 2);

for i = 1 : sdparams.maxit
    [alpha, x] = StepSize(fun, x, -x.g, 1, stepSizeParam);
    if norm(x.g, Inf) <= sdparams.toler * (1+abs(x.f))
        inform.status = 1;
        inform.iter = iter;
        return;
    end
end

inform.status = 0;
inform.iter = sdparams.maxit;
return;
```

Results are show below:

```
** Fletcher-Reeves CG on xpowsing
Success: 65 steps taken

```

```
  Ending value: 4.293e-07 ; No. function evaluations: 205; No.
     gradient evaluations 88
  Norm of ending gradient: 7.732e-06


** PR+ CG on xpowsing
Success: 59 steps taken

  Ending value: 9.506e-07 ; No. function evaluations: 182; No.
     gradient evaluations 86
  Norm of ending gradient: 9.473e-06


** Steepest Descent on xpowsing
CONVERGENCE FAILURE: 10000 steps were taken without
gradient size decreasing below    1e-05.

  Ending value: 7.69e-06 ; No. function evaluations: 35002; No.
      gradient evaluations 15001
  Norm of ending gradient: 0.000135
```