Contents

```
function [uout]=iterative_solve(uin,RHS,maxiter,invA,invB,A,B,a,b,c,Nx,Ny,itype,omg)
```

Point Gauss-Seidel + MG (red-black+SOR)

```
  if (itype==1)
    uout(1:Ny,1:Nx)=0;
    utemp(1:Ny,1:Nx,1:2)=0;
    utemp(1:Ny,1:Nx,1)=uin(1:Ny,1:Nx);
    utemp(1,:,2)=utemp(1,:,1);
    utemp(Ny,:,2)=utemp(Ny,:,1);
    utemp(:,1,2)=utemp(:,1,1);
    utemp(:,Nx,2)=utemp(:,Nx,1);

    for iter=1:maxiter
       for i=2:Nx-1
          for j=2:Ny-1
            if (mod(i+j,2)~=0)
                utemp(j,i,2)=RHS(j,i)*(1/b)-(a/b)*(utemp(j,i-1,1)+utemp(j,i+1,1))...
                                            -(c/b)*(utemp(j-1,i,1)+utemp(j+1,i,1));
            end
          end
       end

       for i=2:Nx-1
         for j=2:Ny-1
            if (mod(i+j,2)==0)
               utemp(j,i,2)=RHS(j,i)*(1/b)-(a/b)*(utemp(j,i-1,2)+utemp(j,i+1,2))...
                                          -(c/b)*(utemp(j-1,i,2)+utemp(j+1,i,2));
            end
         end
       end
       %SOR
       utemp(2:Ny-1,2:Nx-1,2)=utemp(2:Ny-1,2:Nx-1,2)*omg+utemp(2:Ny-1,2:Nx-1,2)*(1-omg);
       utemp(1:Ny,1:Nx,1)=utemp(1:Ny,1:Nx,2);
    end
    uout(1:Ny,1:Nx)=utemp(1:Ny,1:Nx,2);%utemp copied with boundary values
  end
```

```
Not enough input arguments.
```

```
Error in iterative_solve (line 4)
  if (itype==1)
```

Point Gauss-Seidel + MG (normal)

```
if (itype==2)
  utemp(1:Ny,1:Nx,1:2)=0;
  utemp(1:Ny,1:Nx,1  )=uin(1:Ny,1:Nx);
  utemp(1   ,:   ,2  )=utemp(1 ,: ,1);
  utemp(Ny  ,:   ,2  )=utemp(Ny,: ,1);
  utemp(:   ,1   ,2  )=utemp(: ,1 ,1);
  utemp(:   ,Nx  ,2  )=utemp(: ,Nx,1);
  uout(1:Ny,1:Nx     )=0;
  for iter=1:maxiter
     for i=2:Nx-1
       for j=2:Ny-1
             utemp(j,i,2)=RHS(j,i)*(1/b)-(a/b)*(utemp(j,i-1,2)+utemp(j,i+1,1))...
                                        -(c/b)*(utemp(j-1,i,2)+utemp(j+1,i,1));
       end
     end
     %SOR
     utemp(2:Ny-1,2:Nx-1,2)=utemp(2:Ny-1,2:Nx-1,2)*omg+utemp(2:Ny-1,2:Nx-1,2)*(1-omg);

     utemp(1:Ny,1:Nx,1)=utemp(1:Ny,1:Nx,2);
  end
  uout(1:Ny,1:Nx)=utemp(1:Ny,1:Nx,2);%utemp copied with boundary values

end
```

Line-SOR ADI + MG

```
if (itype==3)
    utemp(1:Ny,1:Nx,1:3)=0;
    utemp(1:Ny,1:Nx,1  )=uin(1:Ny,1:Nx);
    utemp(1   ,:   ,2  )=utemp(1 ,: ,1);
    utemp(Ny  ,:   ,2  )=utemp(Ny,: ,1);
    utemp(:   ,1   ,2  )=utemp(: ,1 ,1);
    utemp(:   ,Nx  ,2  )=utemp(: ,Nx,1);

    utemp(1   ,:   ,3  )=utemp(1 ,: ,1);
    utemp(Ny  ,:   ,3  )=utemp(Ny,: ,1);
    utemp(:   ,1   ,3  )=utemp(: ,1 ,1);
    utemp(:   ,Nx  ,3  )=utemp(: ,Nx,1);

  for iter=1:maxiter
```

```
   for j=2:Ny-1
    BCmatx(1,1:Nx-2)=0;
    BCmatx(1,   1)=-a*utemp(j ,1 ,1);
    BCmatx(1,Nx-2)=-a*utemp(j ,Nx,1);
    Atemp(1:Nx-2,1)=RHS(j,2:Nx-1)-c*(utemp(j+1,2:Nx-1,1)...
                    +utemp(j-1,2:Nx-1,2))+BCmatx(1,1:Nx-2);
    utemp(j,2:Nx-1,2)=mtimes(invB,Atemp);
    %utemp(j,2:Nx-1,2)=B\Atemp(1:Nx-2,1);
    %utemp(j,2:Nx-1,2)=tridiag(b*ones(Nx-2,1),a*ones(Nx-2,1),...
    %            a*ones(Nx-2,1),Atemp(1:Nx-2,1));
   end

   for i=2:Nx-1
    BCmaty(1:Ny-2,1)=0;
    BCmaty(1,   1)=-c*utemp(1 ,i,1);
    BCmaty(Ny-2,1)=-c*utemp(Ny,i,1);
    Btemp(1:Ny-2,1)=RHS(2:Ny-1,i)-a*(utemp(2:Ny-1,i-1,3)...
                    +utemp(2:Ny-1,i+1,2))+BCmaty(1:Ny-2,1);
    utemp(2:Ny-1,i,3)=mtimes(invA,RHS(2:Ny-1,i)...
                -a*(utemp(2:Ny-1,i-1,3)+utemp(2:Ny-1,i+1,2))+BCmaty(1:Ny-2,1
    %utemp(2:Ny-1,i,3)=A\(RHS(2:Ny-1,i)-a*(utemp(2:Ny-1,i-1,1)...
    %                  +utemp(2:Ny-1,i+1,1))+BCmaty(1:Ny-2,1));
    %utemp(2:Ny-1,i,3)=tridiag(b*ones(Nx-2,1),c*ones(Nx-2,1),c*ones(Nx-2,1),Btemp(1:Ny-2
   end

   %SOR
   utemp(2:Ny-1,2:Nx-1,3)=utemp(2:Ny-1,2:Nx-1,1)*(1-omg)+utemp(2:Ny-1,2:Nx-1,3)*omg;
   utemp(1:Ny,1:Nx,1)=utemp(1:Ny,1:Nx,3);
  end
  uout(1:Ny,1:Nx)=utemp(1:Ny,1:Nx,3);%utemp copied with boundary values
 end

end

function [uout]=fine_to_coarse(uin,RHS,maxiter,invA,invB,A,B,a,b,c,nx,ny,itype,omg,Nlvlmax,N

%Fine to coarse
for n=Nlvlmin:Nlvlmax
fprintf('na=%d \n',n)
%Solve for Lu=R iteratively "maxiter" times
uout{n}=iterative_solve(uin{n},RHS{n},maxiter,invA{n},invB{n},A{n},B{n},a(n),b(n),c(n),nx(n)
%Compute the residual from the smoothened solution
eps{n}=residual(uout{n},RHS{n},a(n),b(n),c(n),nx(n),ny(n));
%Define the new RHS for the next level
if(n~=Nlvlmax)
epsnew{n+1}=restriction(eps{n},nx(n),ny(n));
```

```
RHS{n+1}=-epsnew{n+1};
uin{n+1}(1:ny(n+1),1:nx(n+1))=0;
end
end


end
```