# *EE 257 Homework 4 Solutions by Ann Chen*

## *Problem 1*

Parallel results

```
 At loop:            3
 At loop:            4
 At loop:            7
 At loop:            8
 At loop:            1
 At loop:            2
 At loop:            9
 At loop:           10
 At loop:            5
 At loop:            6
```

Non Parallel results:

```
 At loop:            1
 At loop:            2
 At loop:            3
 At loop:            4
 At loop:            5
 At loop:            6
 At loop:            7
 At loop:            8
 At loop:            9
 At loop:           10
```

## *Problem 2*

```
Parallel:
Running the loops takes   0.1523438      seconds.
 at row:           1 ,at col:           1 ,the value is  0.674276436940251
 at row:           2 ,at col:           2 ,the value is  0.615143274086586
 at row:           3 ,at col:           3 ,the value is  0.373774997959284
 at row:           4 ,at col:           4 ,the value is  0.703334289873379
 at row:           5 ,at col:           5 ,the value is  0.521546311888681
 at row:           6 ,at col:           6 ,the value is  0.517963762246180
 at row:           7 ,at col:           7 ,the value is  0.703778145332345
 at row:           8 ,at col:           8 ,the value is  0.379409618133927
 at row:           9 ,at col:           9 ,the value is  0.612775344955913
 at row:          10 ,at col:          10 ,the value is  0.675627578895218
Non-Parallel:
 Running the loops takes    1.039063      seconds.
 at row:           1 ,at col:           1 ,the value is  0.674276436940251
 at row:           2 ,at col:           2 ,the value is  0.615143274086586
 at row:           3 ,at col:           3 ,the value is  0.373774997959284
 at row:           4 ,at col:           4 ,the value is  0.703334289873379
 at row:           5 ,at col:           5 ,the value is  0.521546311888681
 at row:           6 ,at col:           6 ,the value is  0.517963762246180
 at row:           7 ,at col:           7 ,the value is  0.703778145332345
 at row:           8 ,at col:           8 ,the value is  0.379409618133927
 at row:           9 ,at col:           9 ,the value is  0.612775344955913
 at row:          10 ,at col:          10 ,the value is  0.675627578895218
```

## *Problem 3*

```
Parallel:
 Running the loops takes    2.3125000      seconds.

Non-Parallel:
 Running the loops takes    17.195313      seconds.
```

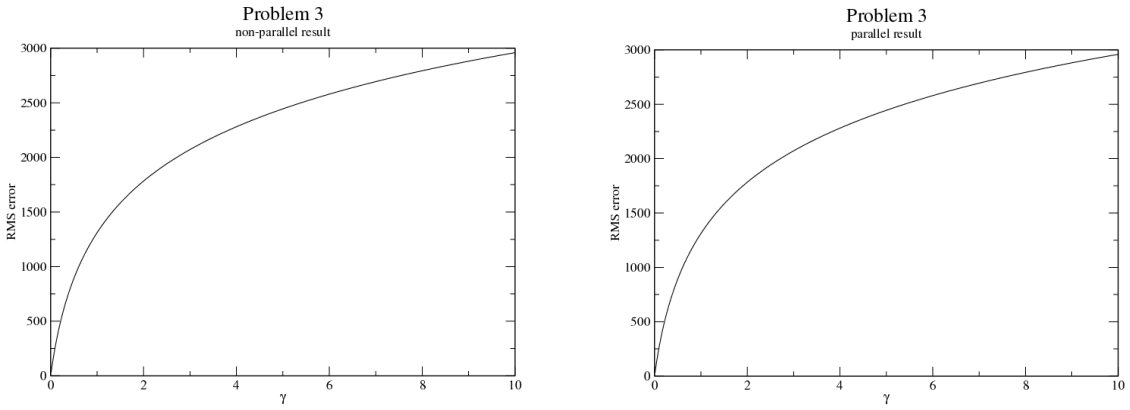


Fig. 1 γ vs. RMS error, as we can see, both parallel and non-parallel code lead to the same results.

## *Problem 4*

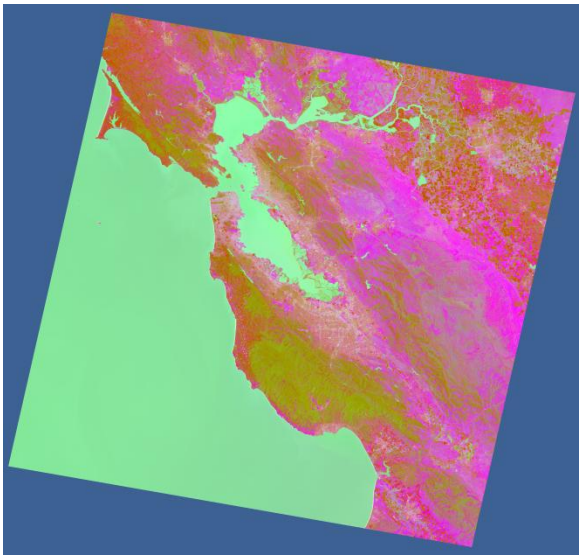

Fig. 2 the PCA image using the largest three PCA components

```
Parallel:
image processing time is:   11.02344

Non Parallel:
image processing time is:   41.88281
```

Note: Speed test was run on a computer with 8 cores.

*FORTRAN 90 code for problem 4*

```
program principal
  use omp_lib
  implicit none   ! this statement forces all variables to the defined

  ! Principal analysis of a 7 channel image
  ! write the result to 7 output image

  !! DECLARE VARIABLES
  character*80 :: fname_in   ! string buffers of length 80
  ! characters
  character*80 ::out1,out2,out3,out4,out5,out6,out7
  integer :: i,n,num  ! counter variable
  real*8,allocatable::input(:,:)  !size n*num
  real*8,allocatable::cv(:,:) !size num*num
  real,allocatable::V(:,:),s(:)!eigen values and eigen vectors
  real*8,allocatable::im(:,:) !size n*num
  real::t0,t1

  !! READ COMMAND LINE INPUTS
  if (iargc().lt.7) then
     ! make sure the user inputs a file name, otherwise print proper usage
     print *, 'USAGE: principal out1 out2 out3 out4 out5 out6 out7'
     print *, ' Principal analysis of a 7 channel image '
     stop    ! stop if number of inputs is incorrect
  else
     call getarg(1,out1)     ! read output file name
     call getarg(2,out2)
     call getarg(3,out3)
     call getarg(4,out4)
     call getarg(5,out5)
     call getarg(6,out6)
     call getarg(7,out7)
  endif

  !!Read input file
  n=7875*7518
  num=7 !number of the channels
  allocate(input(n,num))
  fname_in='chan1.raw'
  call readdata(n,fname_in,input(:,1))
  fname_in='chan2.raw'
  call readdata(n,fname_in,input(:,2))
  fname_in='chan3.raw'
  call readdata(n,fname_in,input(:,3))
  fname_in='chan4.raw'
  call readdata(n,fname_in,input(:,4))
  fname_in='chan5.raw'
  call readdata(n,fname_in,input(:,5))
  fname_in='chan6.raw'
  call readdata(n,fname_in,input(:,6))
  fname_in='chan7.raw'
  call readdata(n,fname_in,input(:,7))

  t0=secnds(0.0)
  !!Calculate covariance
```

```
  allocate(cv(num,num))
  allocate(V(num,num))
  allocate(s(num))
  call computecv(n,num,input,cv)
  !!Principal analysis
  call eigen(num,real(cv,kind=4),s,V)
  do i=1,num
     if(V(1,i)<0)then
        V(:,i)=V(:,i)*-1
     endif
  enddo
  !!Compute 7 output images
  allocate(im(n,num))
  call imagecomp(n,num,real(V(:,:),kind=8),input,im)

  !!Scale the 7 images with mean 140 deviation 60
  call scaleimage(n,num,im)

  t1=secnds(t0)
  print *,'image processing time is:',t1

  !! Write output files
  call writedata(n,im(:,1),out1)
  call writedata(n,im(:,2),out2)
  call writedata(n,im(:,3),out3)
  call writedata(n,im(:,4),out4)
  call writedata(n,im(:,5),out5)
  call writedata(n,im(:,6),out6)
  call writedata(n,im(:,7),out7)

  deallocate(input,cv,im,V,s)

end program principal

subroutine readdata(n,fname_in,image)
  !read the input file fname, return the image(x,y)
  !the avarage value has been removed
  implicit none
  integer::n,i
  character*80::fname_in
  real*8::image(n),mean
  integer*1::temp(n)

  open(21,file=fname_in,access='direct',recl=1*n)  ! reads n  variables, with
size 1 bytes
  read(21,rec = 1) temp
  close(21)   ! close input file
  do i=1,n
     if(temp(i)>=0)then
        image(i)=real(temp(i),kind=8)
     else
        image(i)=real(temp(i),kind=8)+256.0_8
     endif
  enddo

  mean=sum(image)/real(n,kind=8)
  image=image-mean
```

```
end subroutine readdata

subroutine computecv(n,num,input,cv)
  implicit none
  integer::i,j,n,num
  real*8::input(n,num)
  real*8::cv(num,num)
  !$omp parallel do private(i,j)shared(num,n,cv,input)
  do i=1,num
     do j=1,num
        cv(i,j)=dot_product(input(:,i),input(:,j))/real(n,kind=8)
     enddo
  enddo
  !$omp end paralleldo
end subroutine computecv

subroutine imagecomp(n,num,V,input,image)
  implicit none
  integer::i,j,n,num
  real*8::input(n,num)
  real*8::image(n,num)
  real*8::V(num,num)
  !$omp parallel do private(i,j)shared(image,V,input,n,num)
  do j=1,num
     do i=1,n
        image(i,j)=dot_product(V(:,j),input(i,:))
     enddo
  enddo
  !$omp end paralleldo
end subroutine imagecomp

subroutine scaleimage(n,num,image)
  implicit none
  integer::n,num,i
  real*8::image(n,num)
  real*8::mean,std,k,b
  !$omp parallel do private(i,mean,std,k,b)shared(image,n,num)
  do i=1,num
     mean=sum(image(:,i))/real(n,kind=8)
     std=sqrt(sum((image(:,i)-mean)**2)/real(n,kind=8))
     k=60.0/std
     b=140.0-k*mean
     image(:,i)=k*image(:,i)+b
  enddo
  !$omp end paralleldo
end subroutine scaleimage

subroutine writedata(n,image,fname_out)
  implicit none
  integer::i,n
  character*80::fname_out
  real*8::image(n)
  integer*1::temp(n)

  do i=1,n
     temp(i)=floor(image(i))
```

```
      if(floor(image(i))>255)then
         temp(i)=255
      else if(floor(image(i))<1)then
         temp(i)=1
      endif
   enddo

   open(unit=2,file=fname_out,access='direct',&  ! & says continue to next
line
         form='unformatted',recl=1*n,status ='replace') ! open output file
   write(2,rec=1) temp
   close(2)    ! close input file

end subroutine writedata
```