1 Project 1

1.1 PDE

We get a parabolic equation because of the gradient descent flow

$$\begin{cases} \dfrac{\partial u}{\partial t} = div\left(\dfrac{\nabla u}{\varphi(|\nabla u|)}\right) - \lambda^*(u-f) \\ \dfrac{\partial u}{\partial \overrightarrow{n}}\Big|_{\partial\Omega} = 0 \\ u(x,0) = f \end{cases} \tag{1}$$

1.2 Numerical format

For convenience of writing, we define

$$D_x^{\pm}(u_{i,j}) \triangleq \pm(u_{i\pm1,j,k} - u_{i,j,k}),$$

We proposed a numerical approximation of $|\nabla u|$

$$|\nabla u| \approx |D_h(u^n_{i,j,k}| = \sqrt{D_x^+(u_{i,j,k})^2 + D_y^+(u_{i,j,k})^2 + D_z^+(u_{i,j,k})^2 + \varepsilon}$$

And then, we have

$$div\left(\frac{\nabla u}{|\nabla u|}\right) \approx D_x^-\left(\frac{D_x^+(u^n_{i,j,k})}{|D_h(u^n_{i,j,k})|}\right) + D_y^-\left(\frac{D_y^+(u^n_{i,j,k})}{|D_h(u_{i,j,k})^n)|}\right) \triangleq z^n_{i,j,k} \tag{2}$$

In addition, we discrete time using forward Euler using forward Euler method. Therefore, the numerical format of parabolic equation (1) is obtained as follow

$$u^{n+1}_{i,j,k} = u^n_{i,j,k} + \tau z^n_{i,j,k} - \tau\lambda(u^n_{i,j,k} - u^0_{i,j,k}), \quad \text{where } \tau = t^{n+1} - t^n. \tag{3}$$

1.3 Distribution and communication

Data distribution:

```
if rank == 0:
  # Generate image
  ...
  # Adding Gaussian noise
  ...
  # Data distribution
  nimgsile = nimg[:,:,0:101]
  sendbuf = nimg[:,:,99:200].copy()
  comm.Send(sendbuf, dest=1, tag=11)
  del nimg, sendbuf
else:
  nimgsile = np.empty([200,200,101],dtype=np.float64)
  comm.Recv(nimgsile, source=0, tag=11)
```

Communication:

```
  for t in range(T):
    if rank ==0 and not t%5:
      print(t, 'of ', T)
    # In-process computation
    J = worker(nimgsile, J, dt, lam)
    # Blocking communication
    if rank == 0:
      # rank 0: send before recv
      sendbuf = J[:,:,n2-2].copy()
      comm.Send(sendbuf, dest=1, tag=100)
      recbuf = np.empty(J[:,:,n2-1].shape, dtype=J[:,:,n2-1].dtype)
      comm.Recv(recbuf, source=1, tag=110)
      J[:,:,n2-1] = recbuf
    else:
      # rank 1: recv before send
      recbuf = np.empty(J[:,:,n2-1].shape, dtype=J[:,:,n2-1].dtype)
      comm.Recv(recbuf, source=0, tag=100)
      J[:,:,0] = recbuf
      sendbuf = J[:,:,1].copy()
      comm.Send(sendbuf, dest=0, tag=110)
```

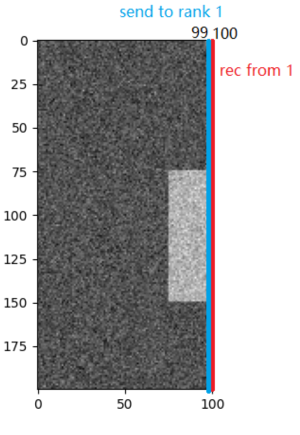


Figure 1: Process 0

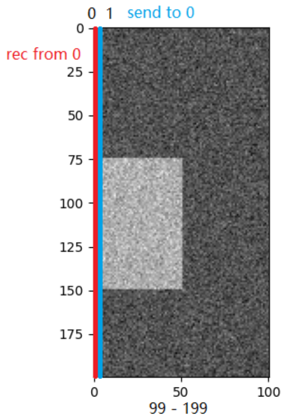


Figure 2: Process 1

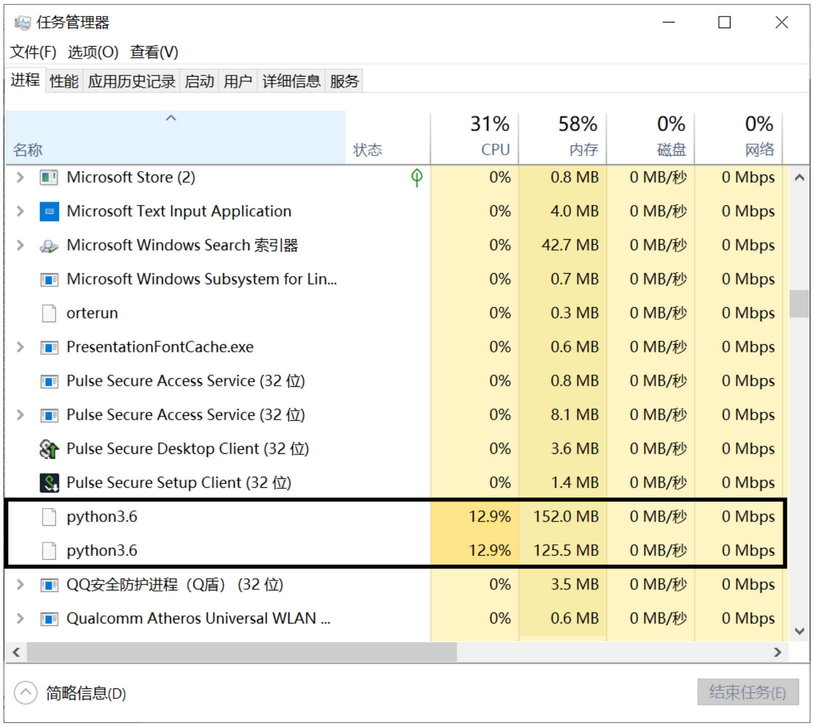
1.4 In-process computation improvement

Original: Matrix operations (faster than for) but increased 5 times RAM.

```
def worker(In, J, dt, lam):
  ep = 1e-4
  m,n,l = J.shape
  # Gradient approximates by forward Euler
  DfJx=J[list(range(1,m))+[m-1],:,:]-J
  DfJy=J[:,list(range(1,n))+[n-1],:]-J
  DfJz=J[:,:,list(range(1,l))+[l-1]]-J
  # \varphi(\nabla u)
  TempDJ=(ep+DfJx*DfJx+DfJy*DfJy+DfJz*DfJz)**(1/2)
  DivJx=DfJx/TempDJ
  DivJy=DfJy/TempDJ
  DivJz=DfJz/TempDJ
```

```
Div=bdx(DivJx,m)+bdy(DivJy,n)+bdz(DivJz,l)
J += dt * Div -dt*lam*(J-In)
return J
```

Improvement: Divided into 8 parts along the x direction, each parts have 25+1 or 25+2 layers. RAM is shown in Figure 3.



名称	状态	31% CPU	58% 内存	0% 磁盘	0% 网络
Microsoft Store (2)		0%	0.8 MB	0 MB/秒	0 Mbps
Microsoft Text Input Application		0%	4.0 MB	0 MB/秒	0 Mbps
Microsoft Windows Search 索引器		0%	42.7 MB	0 MB/秒	0 Mbps
Microsoft Windows Subsystem for Lin...		0%	0.7 MB	0 MB/秒	0 Mbps
orterun		0%	0.3 MB	0 MB/秒	0 Mbps
PresentationFontCache.exe		0%	0.6 MB	0 MB/秒	0 Mbps
Pulse Secure Access Service (32 位)		0%	0.8 MB	0 MB/秒	0 Mbps
Pulse Secure Access Service (32 位)		0%	8.1 MB	0 MB/秒	0 Mbps
Pulse Secure Desktop Client (32 位)		0%	3.6 MB	0 MB/秒	0 Mbps
Pulse Secure Setup Client (32 位)		0%	1.4 MB	0 MB/秒	0 Mbps
python3.6		12.9%	152.0 MB	0 MB/秒	0 Mbps
python3.6		12.9%	125.5 MB	0 MB/秒	0 Mbps
QQ安全防护进程（Q盾） (32 位)		0%	3.5 MB	0 MB/秒	0 Mbps
Qualcomm Atheros Universal WLAN ...		0%	0.6 MB	0 MB/秒	0 Mbps

Figure 3: RAM Improvement

1.5 Results

Results collection and presentation:

```
if rank ==0:
  # Process0 collect data and plt
  deimg = np.empty([nx,ny,nz], dtype = np.float64)
  deimg[:,:,0:100] = J[:,:,0:100]
  recbuf = np.empty([200,200,100],np.float64)
  comm.Recv(recbuf, source=1, tag=20)
  deimg[:,:,100:200] = recbuf
  plt.figure()
  plt.imshow(deimg[100,:,:],"gray")
  ...
else:
  sendbuf = J[:,:,1:101].copy()
  comm.Send(sendbuf, dest=0, tag=20)
```

Results is shown Figure 4-6. (Code see **TV3d2.py**)

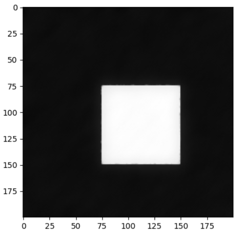


Figure 4: y-z Section

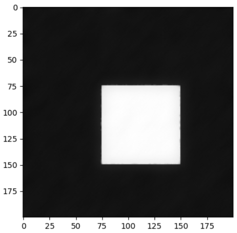


Figure 5: x-z Section

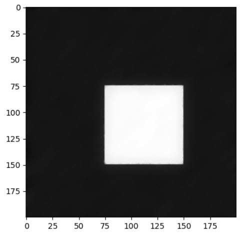


Figure 6: x-y Section