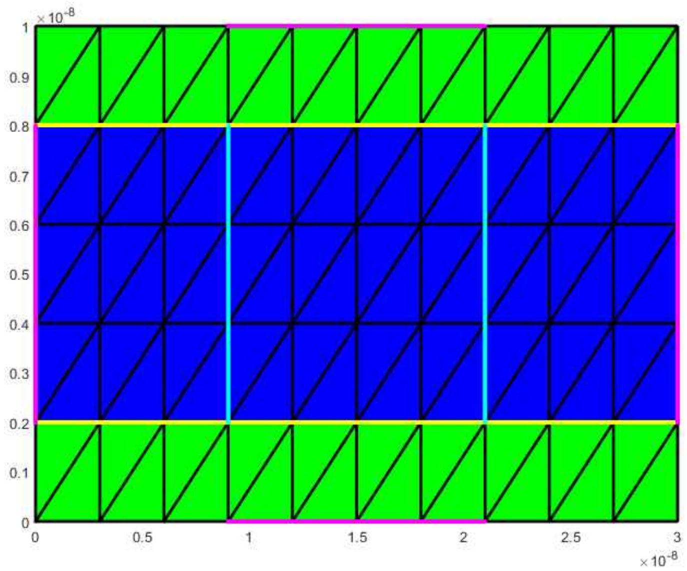
HW13

20211119 박 건 호

Double-Gate Structure

Structure	Region & Interface & Contact
	Green region : Oxide region Blue region : Silicon region Yellow line : Si-Oxide interface magenta line : Gate, Source, Drain Contact Cyan line : Si interface (First Cyan line: Left = N-type , Right = P-type Second Cyan line:Right : N-type)

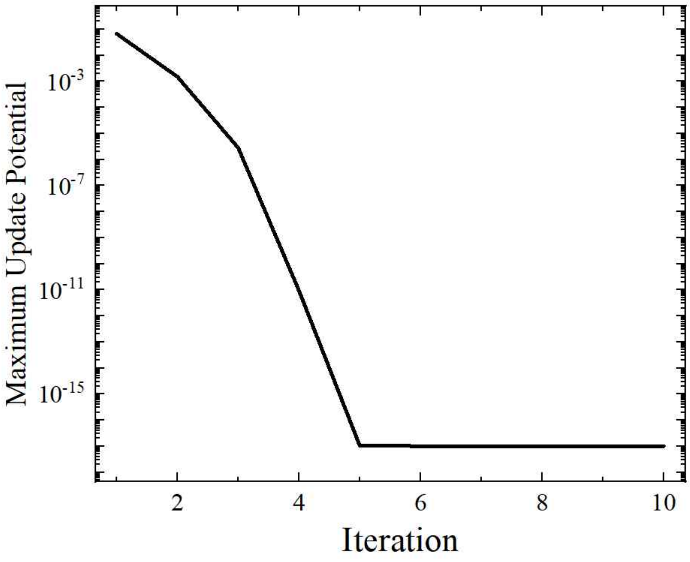
Total Vertex : 66 Point

Jacobian Table size : 176 * 176 matrix

Equilibrium Nonlinear Poisson (n,p를 다음과 같이 넣은 상태)

$$res_e = elec - n_i(\exp(\frac{\phi}{V_T})) \quad res_h = hole - n_i(\exp(\frac{-\phi}{V_T}))$$

지난 HW10의 과제인 Nonlinear Poisson 코드에 대해서 바뀐 Device에서도 수렴하는지 확인하였습니다. 실제 이 Device에는 도핑을 고려해주고 있으며, Si 내의 region을 따로 나눠서 생각하지 않고 Table을 활용하여 if 문으로 변경하였습니다.
Convergence에 대한 결과는 다음과 같습니다.



Interface의 노드를 포함한 총 88개의 Phi의 update에서 모두 다음과 같이 쿼드라틱하게 수렴하는 모습을 확인하였습니다.
따라서, Nonlinear Poisson은 수렴하는 것을 확인하였습니다.

Iteration을 보기에 앞서서 확인했던 부분은 phi의 초기해에 대한 값입니다. 이전까지 도핑을 고려한 laplacian을 풀기는 했지만, 도핑의 값이 한 device에서 크게 변경될 때 laplacian에서 특정 phi의 값이 크게 증가하여 수렴하지 못하는 문제가 있었고, 이 부분을 해소하기 위해서 계산전자공학의 책을 참고하여 phi의 값을 도핑농도로부터 구하는 수식을 활용하여 초기해를 줬습니다.

Drift-Diffusion

이전 수업까지 Jn,Jp에 대한 내용을 공부했고, Scharfetter-gummel scheme에 대한 내용까지 공부하였습니다. 이를 바탕으로 베르누이 수식을 활용한 Jn,Jp를 공식을 정리하였습니다.

$$J_n = q\mu_n V_T \frac{A_{ji}}{l_{ji}}\left(n_j B\left(\frac{\nabla\phi}{Vt}\right) - n_i B\left(\frac{-\nabla\phi}{Vt}\right)\right)$$

$$J_p = -q\mu_p V_T \frac{A_{ji}}{l_{ji}}\left(p_j B\left(\frac{-\nabla\phi}{Vt}\right) - p_i B\left(\frac{\nabla\phi}{Vt}\right)\right)$$

해당 수식을 각각 n, p potential을 통해서 미분해주면 Jacobian을 형성할 수 있고, 실제 Potential과 동일하게 Vertex index를 Table에서 Find 하여 Jacobian의 행과 열을 설정하였습니다.
베르누이 수식은 matlab의 Function을 통해서 제작하였고 그 수식은 다음의 사진과 같습니다.

```
function y = Ber(x)
%Bernoulli function
    if(abs(x)<0.02502)
        y = ( 1.0-(x)/2.0 +(x)^2/12.0*(1.0-(x)^2/60.0*(1.0-(x)^2/42.0)) ) ;
    elseif(abs(x)<0.15)
        y = ( 1.0-(x)/2.0+(x)^2/12.0*(1.0-(x)^2/60.0*(1.0-(x)^2/42.0*(1-(x)^2/40*(1-0.025252525252525252525*(x)^2)))))
    else
        y = x/(exp(x)-1);
    end
end
```

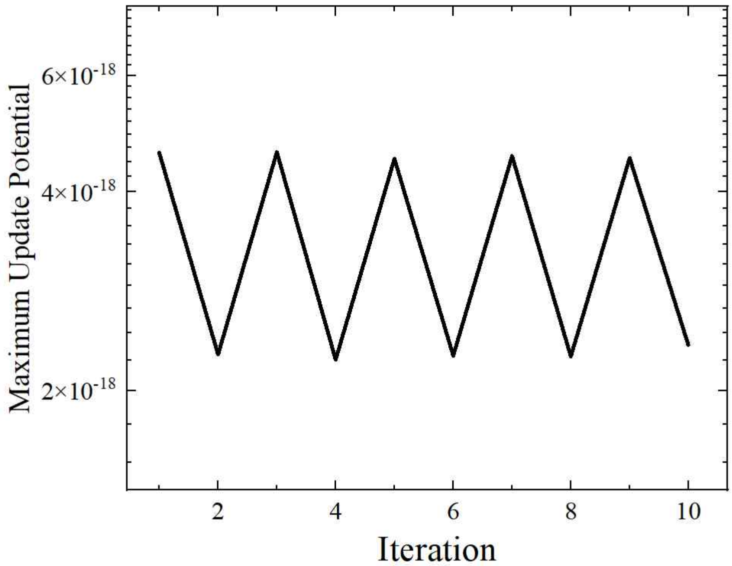
```
function y = Ber_d(x)
%Derivative of Bernoulli function
    if(abs(x)<0.02502)
        y = (-0.5 + (x)/6.0*(1.0-(x)^2/30.0*(1.0-(x)^2/28.0)) );
    elseif(abs(x)<0.15)
        y = (-0.5 + (x)/6.0*(1.0-(x)^2/30.0*(1.0-(x)^2/28.0*(1-(x)^2/30*(1-0.031565656565656565657*(x)^2)))));
    else
        y = -(exp(x))/(exp(x)-1)^2;
    end
end
```

Youtube 강의를 바탕으로 수식을 작성하였고, Potential의 변화가 작을 때보다 높은 정확도를 위해서 Taylor series로 전개하여 작성하였습니다.

이 식을 활용해서 Coding을 하였고 결과를 확인하니, 준 특이행렬이 발생하였습니다. Scaling이나, Jacobian을 구성하면서 실수한 점을 Debugging이 필요해보입니다.

하지만, 이 경우에서도 수업에서 배운 것처럼, Nonlinear의 해를 받아 DD를 계산한다면 Iteration이 1번째부터 완벽히 수렴해야했고, 이 점을 확인해봤습니다.

DD iteration 총 10회 계산 **(Vg=Vs=Vd=0V)**



모든 노드에서 다음과 같이 첫 iteration부터 1e-18 수준의 update를 확인하였고, elec, hole 역시 모두 수렴하는 모습을 확인하였습니다.

Bias를 ramping 하여도 수렴성을 보였으나, 지속적으로 준 특이행렬이 나옴으로 실제 계산 결과가 옳은지 의문이며, 이를 수정한 뒤에 Bias 조건을 모두 확인하는 것이 옳다라고 판단하였습니다.