Projectes d'Enginyeria Física - 2

Modelling and designing a Paul ion trap

V2024.2

Juan M. Rius

Universitat Politècnica de Catalunya (UPC)

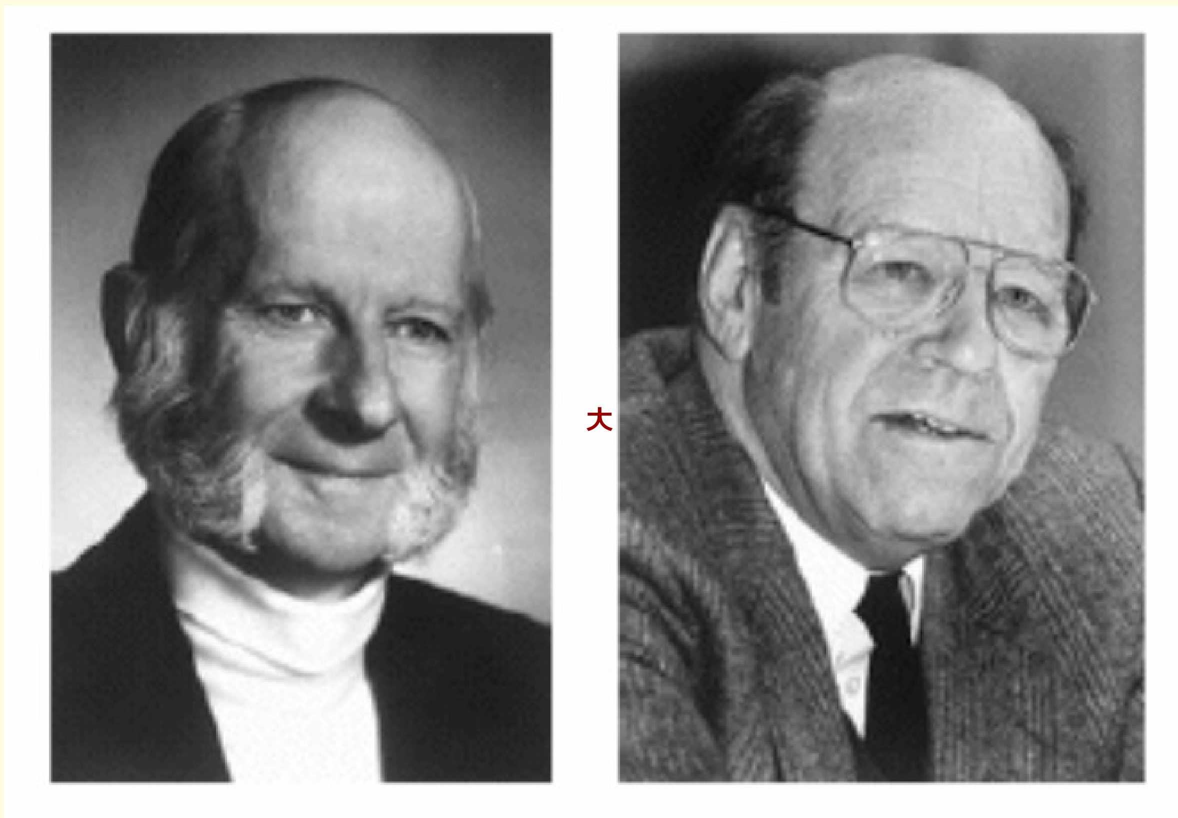
Barcelona, Spain

Contents

- Introduction to the Paul ion trap
- Introduction to Computational Electromagnetics
- Modeling the problem with Integral Equation methods
- Discrete equations with the Method of Moments
- Implementation in 2D: validation with capacitor
- Implementation in 3D: validation with capacitor
- Analyze and design a Paul Trap:
  - Static potential, one ion
  - AC potential, one ion
  - AC potential, multiple ions
  - Design the trap for optimum ion confinement

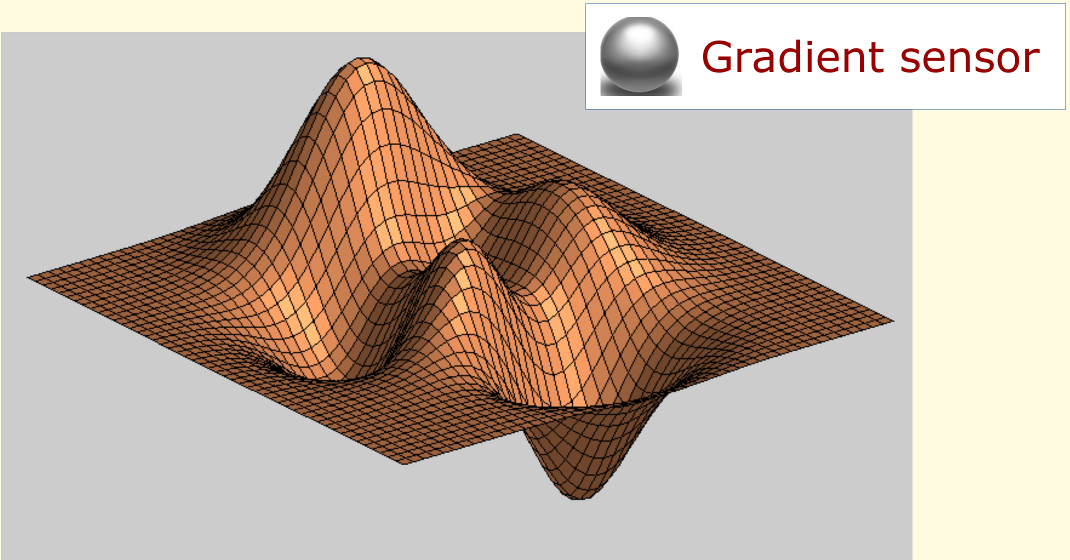
The Paul ion trap

- **Pioneers of ion trapping:**
  Hans Dehmelt and Wolfgang Paul (Nobel price 1989)



The Paul ion trap

The force on ions follows the direction of field: $\vec{E} = -\nabla V$



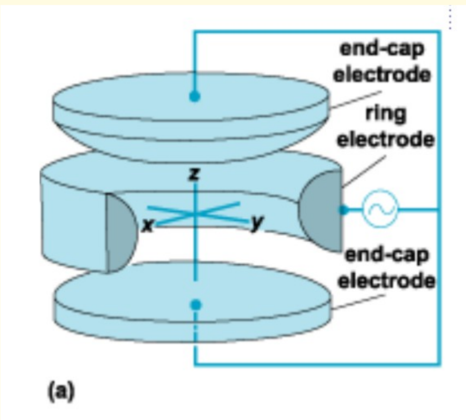
- **It is not possible to have a 3D minimum or maximum of V(r), since**

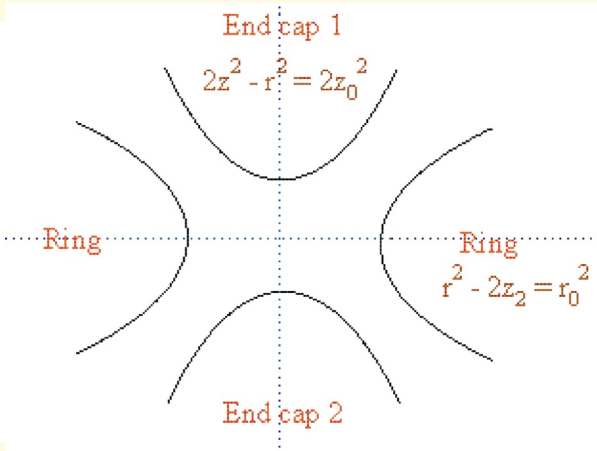
$$\nabla^2 V\left(\vec{r}\right) = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}\right) V\left(\vec{r}\right) = 0$$

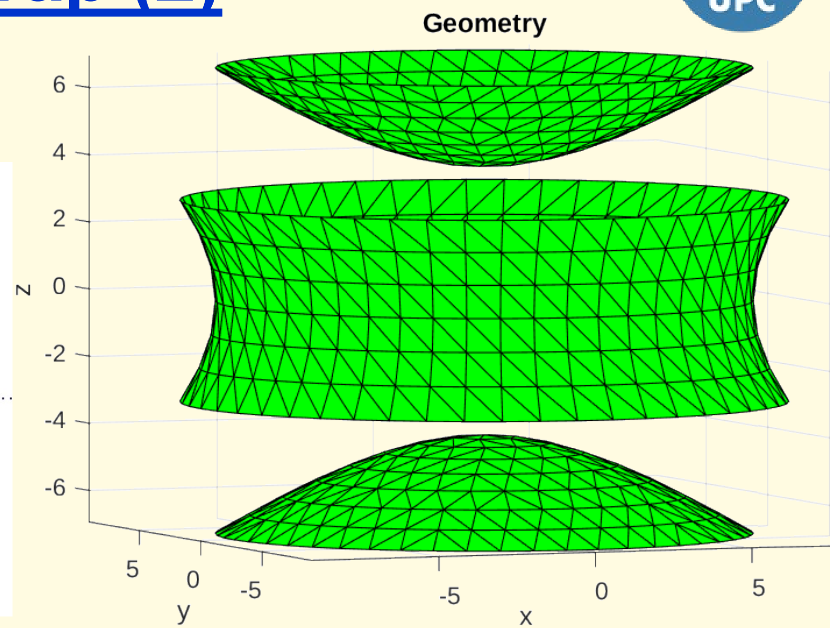
  – **Anyway, it would work only for positive or negative ions**

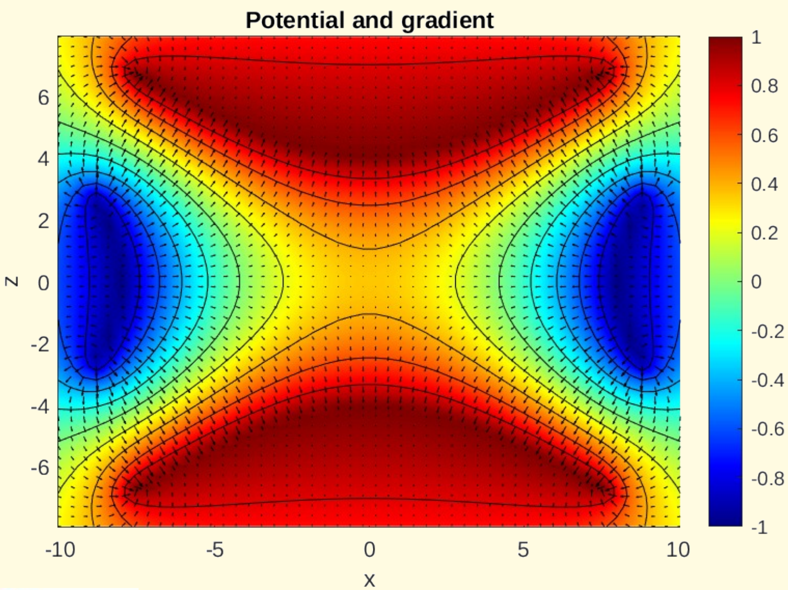
The Paul ion trap (2)

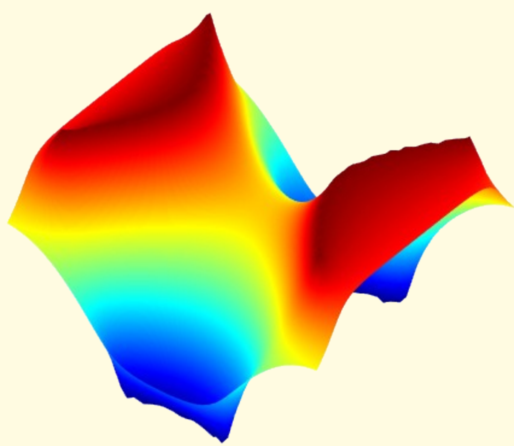
Quadrupole:

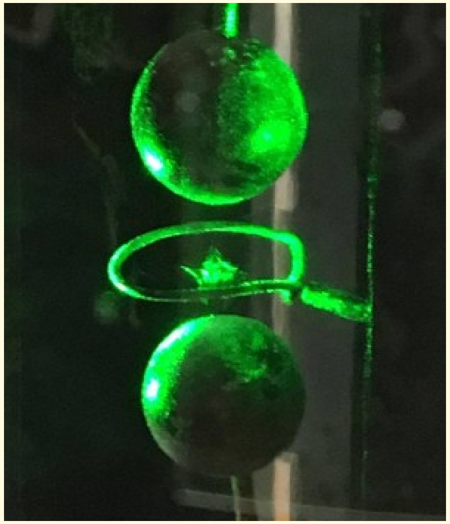






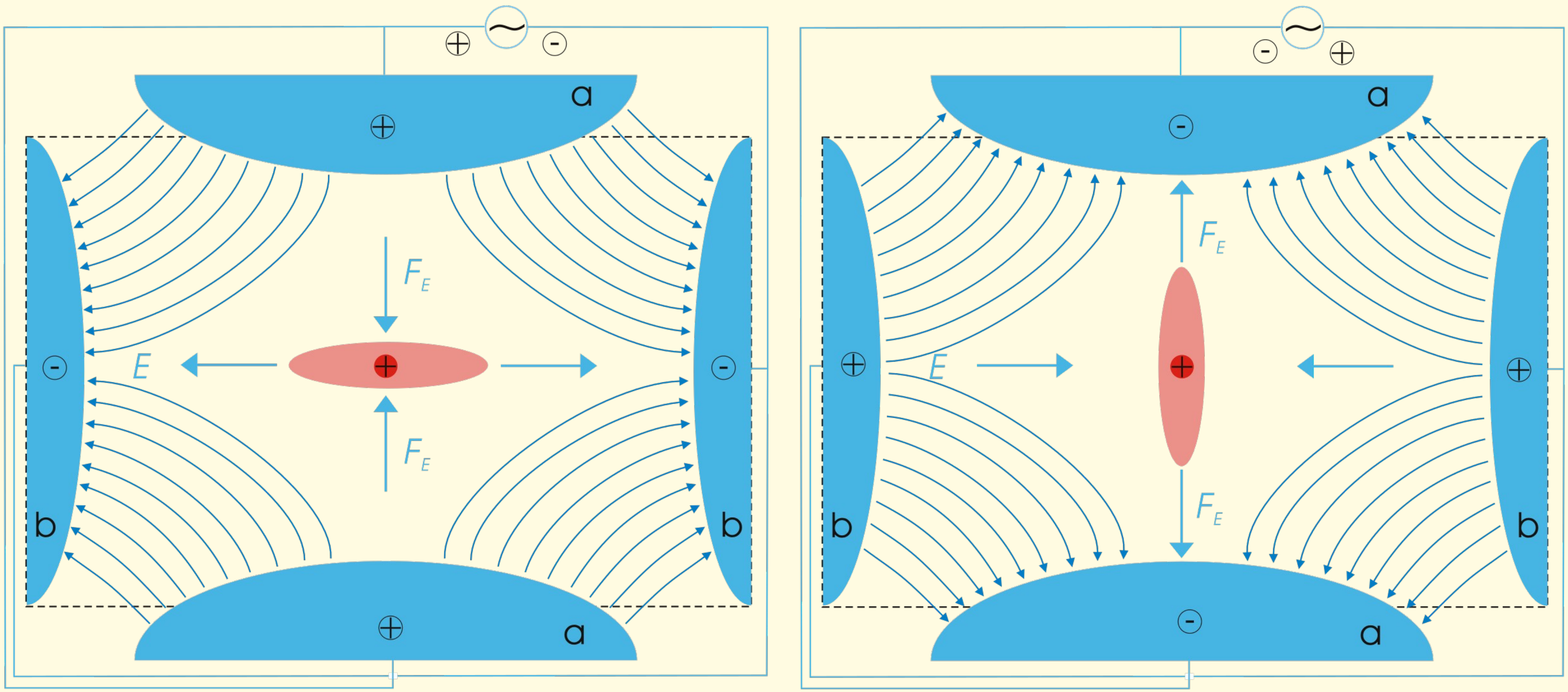






The Paul ion trap (3)

- **Quadrupole with AC potential variation (low freq.)**

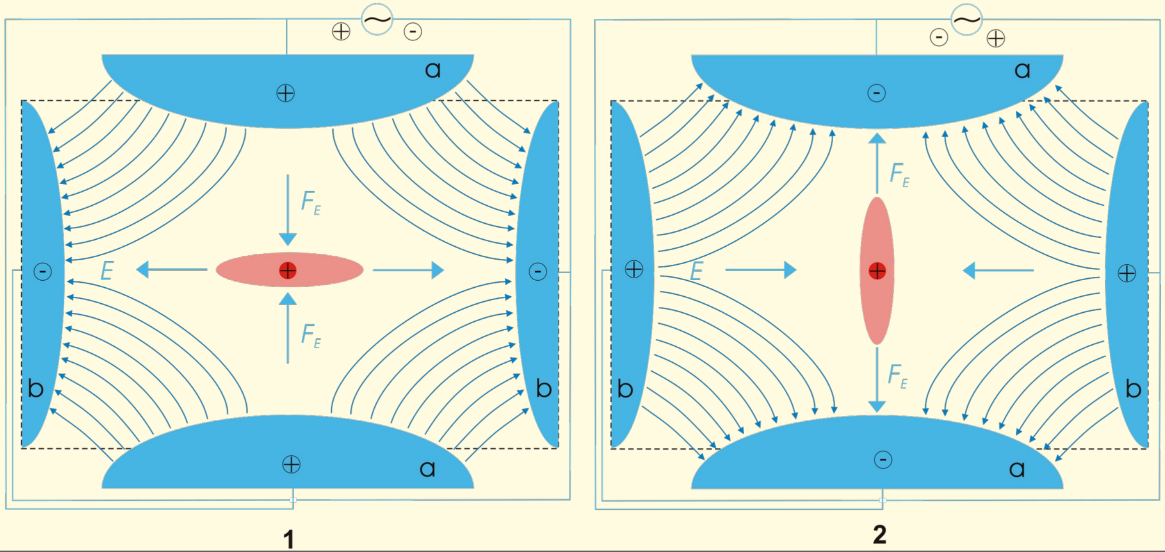


1   2

  - **Now, ions rotate movement direction, following AC changes, and become confined at the center of the trap**

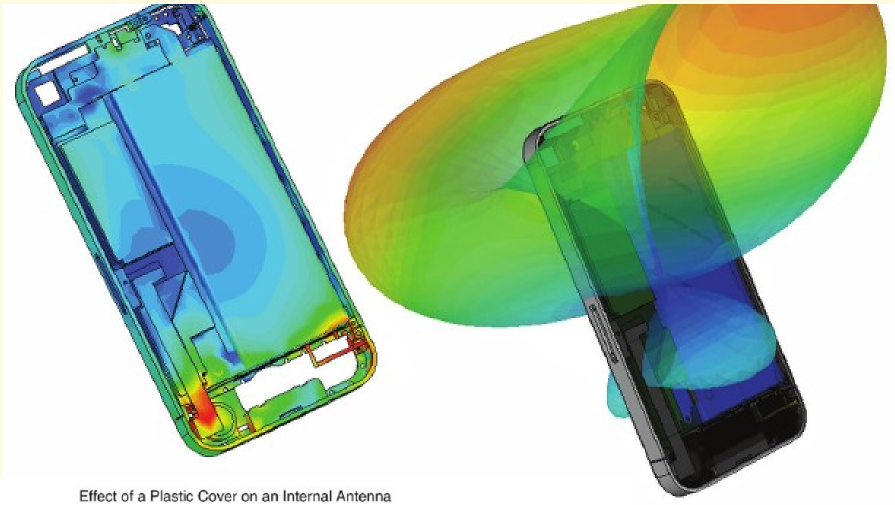
The Paul ion trap (4)

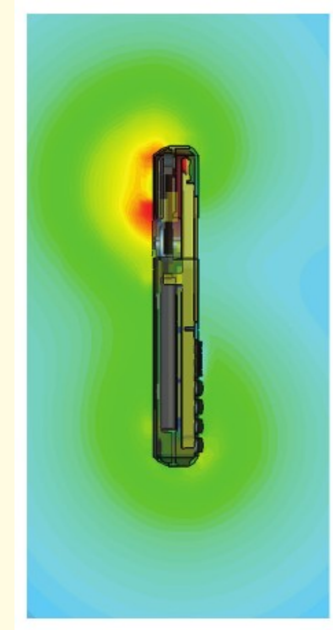
- **Objective:**
  - Develop computer code for computing the potential due to hyperbolic quadrupole.
  - Compute ion trajectories within the trap.
  - Account for AC variation of potential → ion trajectories
  - Design size of the trap, AC frequency, etc… for ion confinement with weights and charge.

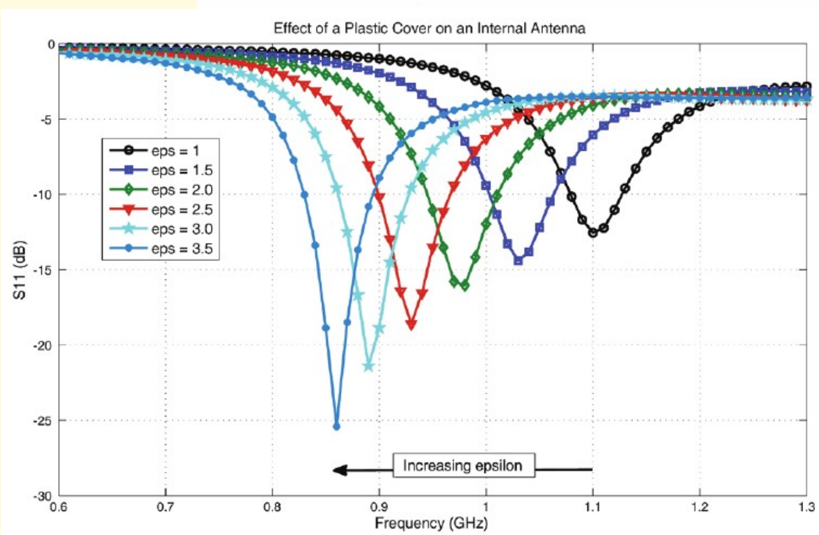


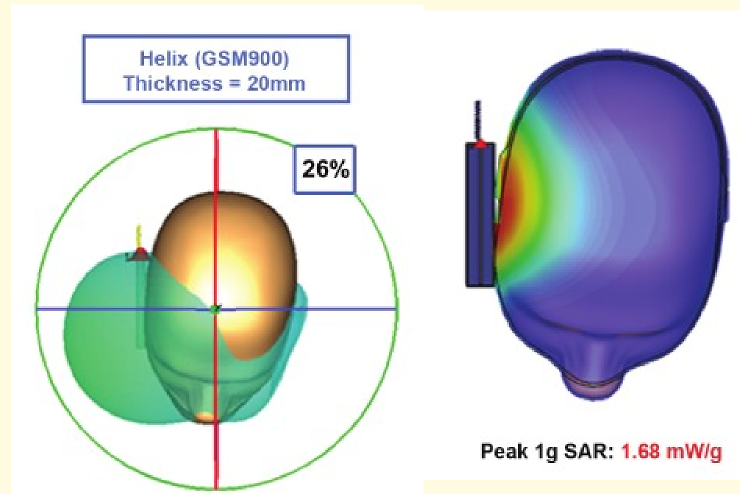
Computational electromagnetics

- Antenna analysis and design using numerical simulation software:









CEM simulation software fundamentals

$$\hat{n} \times \vec{E}^{i}(\vec{r}) = \hat{n} \times \iint_S \left[jk\eta\, G(\vec{r}\,' - \vec{r})\, \vec{J}_s(\vec{r}\,') + \frac{\eta}{jk} \nabla' G(\vec{r}\,' - \vec{r})\, \nabla' \cdot \vec{J}_s(\vec{r}\,') \right] ds$$

Diff. or integ. equations

$$G = \frac{e^{-jk|\vec{r}-\vec{r}\,'|}}{4\pi|\vec{r}-\vec{r}\,'|}$$

Singular!

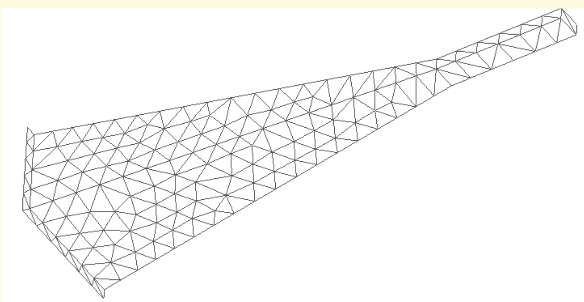
Discretization

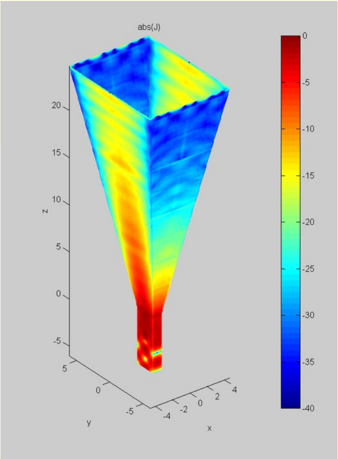
Linear system of equations

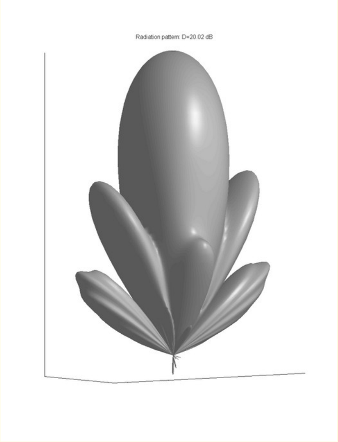
Full matrix with **millions of unkowns**!

Fast Solver

Solution $\vec{J}(\vec{r}\,')$



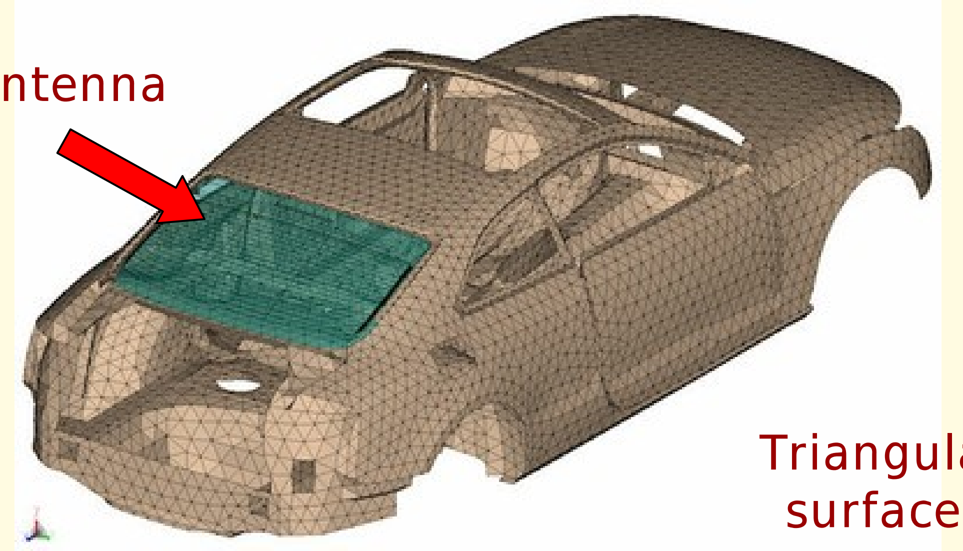




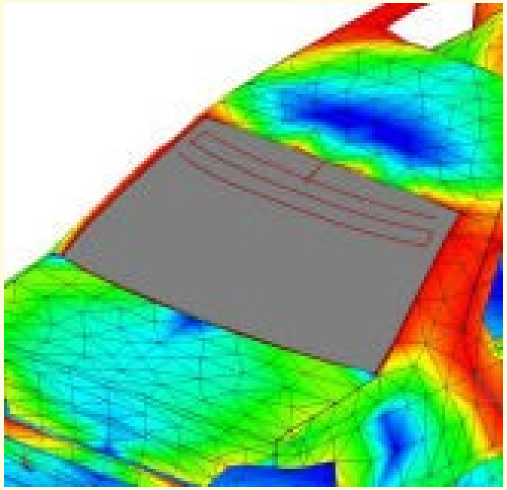
Antenna parameters: input impedance, radiation pattern, etc.

Geometry model: triangular mesh

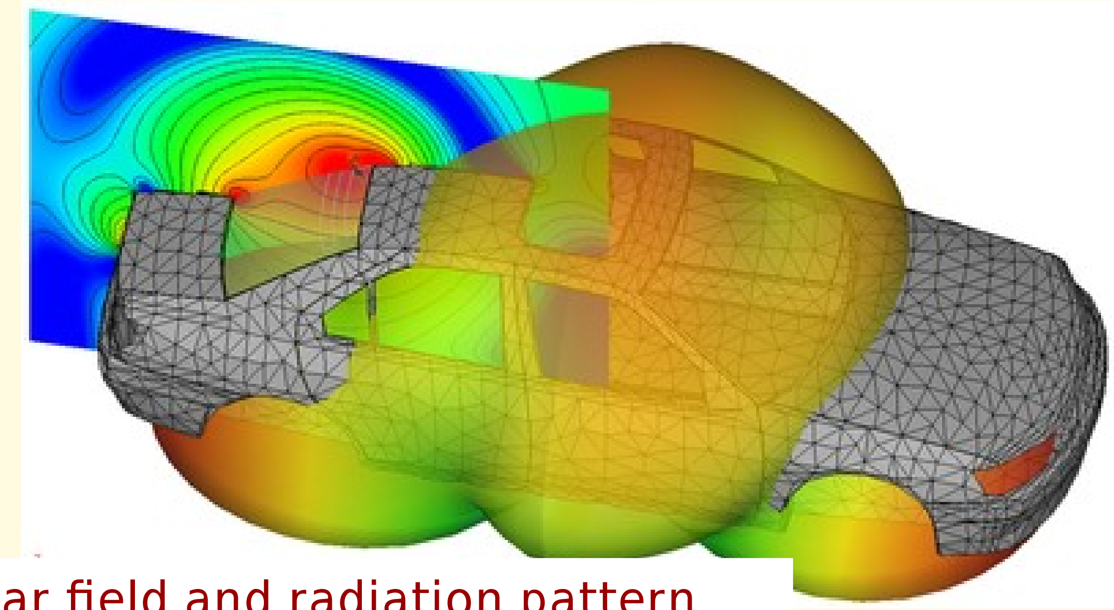
Antenna



Triangular mesh surface model



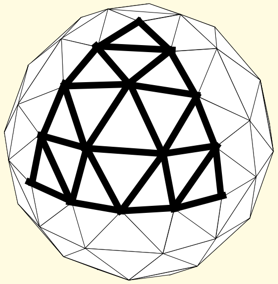
Induced currents



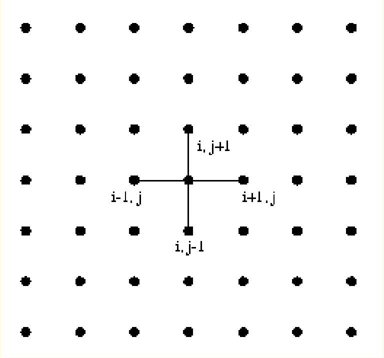
Near field and radiation pattern

Solution steps

- **Formulation of (linear) differential or integral equations**
  - Time or frequency domain
  - Examples: Maxwell's or wave equations, EFIE, MFIE,….
- **Discretization of equations into a linear system**
  - Method of moments, finite element method, etc..



- **Discretization of boundary conditions**
  - Discretize surfaces into triangle meshes….
  - Set numerical boundary condition (samples, min sq error, ...)
- **Set mesh truncation conditions**
  - Only for differential equations, not necessary for IE
  - Finite mesh: a different equation must be set at mesh truncation nodes



Sources of error

- **Simplifications and approximations in the equations:**
  - Problem modeling
  - Material parameters
- **Discretization:**
  - Equations into linear system (projection into finite-dimensional space)
  - Numerical integral computation (often with singularities)
  - Boundary conditions surfaces into meshes
  - Boundary conditions into numbers
  - Truncation mesh conditions are always approximate
- **Linear system solution**
  - Truncation of real numbers into a finite-legnth word
  - Iterative methods
  - Fast solvers for huge linear systems: always involve approximations

There are many things than can go wrong!!

Integral equation methods

- **Transform DE into IE with Green's function:**

$$\phi(\vec{r}) = \mathcal{L} f(\vec{r})$$

$\mathcal{L}$=linear operator, $\phi(\vec{r})$ =field/potential, $f(\vec{r})$= sources
(invariant)

$$G(\vec{r}) = \mathcal{L}\delta(\vec{r})$$

G is the impulse response or **Green's function**

$$\phi(\vec{r}) = \int_S f(\vec{r}\,')G(\vec{r}-\vec{r}\,')d\vec{r}\,'$$

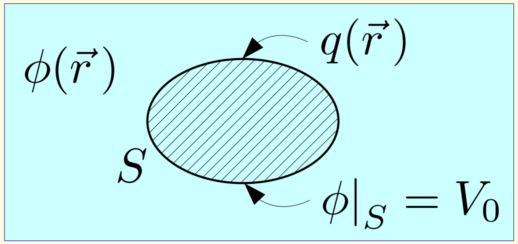
- **Set boundary condition:** $\phi(\vec{r})|_S = \phi_0$

$$\left. \int_S f(\vec{r}\,')G(\vec{r}-\vec{r}\,')d\vec{r}\,' \right|_S = \phi_0$$

Solution domain on S instead of whole volume
⇒ **discretize only on S** ⇒ much less unknowns

Integral equation for electrostatics (3D)

- **Electrostatic potential (3D):**
  - **Poisson's equation:** $\nabla^2 \phi(\vec{r}) = -\dfrac{q(\vec{r})}{\varepsilon}$



  - **Green's function:** $\nabla^2 G(\vec{r}) = -\delta(\vec{r})$ $\qquad G = \dfrac{1}{4\pi\varepsilon|\vec{r}\,|}$

$$\phi(\vec{r}\,) = \int_S q(\vec{r}\,')\frac{1}{4\pi\varepsilon|\vec{r} - \vec{r}\,'|}d\vec{r}\,'$$

  - **Boundary condition:**

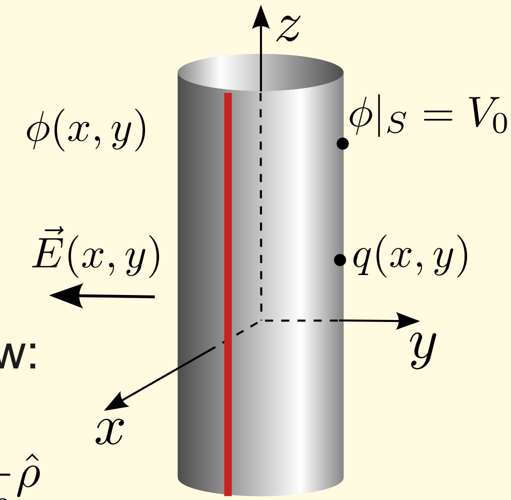
$$\phi(\vec{r}\,)|_S = V_0$$

  - **Integral equation:**

$$\boxed{\int_S q(\vec{r}\,')\frac{1}{4\pi\varepsilon|\vec{r} - \vec{r}\,'|}d\vec{r}\,' \Big|_S = V_0}$$

Integral equation for electrostatics (2D)

- **Potential Green's function in 2D:**



  - Green's function: Potential due to a **line of charge**
  - First we compute *E* field due to a point source $q_0=1$,
    E must be $\vec{E}(\rho,\phi) = E_\rho(\rho)\hat{\rho}$ and according to Gauss' law:

Per unit length: $\frac{q_0}{\varepsilon} = \oint \vec{E}\cdot\hat{n}\, dl = 2\pi\rho E_\rho \quad \Rightarrow \quad \vec{E} = \frac{q_0}{2\pi\varepsilon\rho}\hat{\rho}$

  - The potential is such that $\vec{E} = -\nabla\phi = -\frac{\partial}{\partial\rho}\phi(\rho)\hat{\rho} = \frac{q_0}{2\pi\varepsilon\rho}\hat{\rho}$

so, the potential $\phi = \frac{-q_0}{2\pi\varepsilon}\ln\rho$ is the Green's function in 2D $(q_0 = 1)$

$$G = \frac{-1}{2\pi\varepsilon}\ln\rho$$

$$\phi(\vec{\rho})|_C = V_0$$

$$\left.\frac{-1}{2\pi\varepsilon}\int_C q(\vec{\rho}\,')\ln\left(|\vec{\rho}-\vec{\rho}\,'|\right) d\ell'\,\right|_C = V_0$$

Electrostatics integral equation in 2D

Method of Moments (MoM) (1)

- **The most commonly used method to discretize electromagnetic integral equations**
- **Valid for any linear equation (also differential eq.)**

$$\mathcal{L}X\left(\vec{r}\right) = Y\left(\vec{r}\right)$$

$\mathcal{L}$ is a **linear** operator

X is the **unknown** function

Y is a **known** function

it will be discretized as a linear system $[Z][a] = [b]$

$$\frac{-1}{2\pi\varepsilon}\int_C q(\vec{\rho}\,')\ln\left(|\vec{\rho}-\vec{\rho}\,'|\right)d\ell'\bigg|_C = V_0$$

Electrostatics integral equation in 2D

$$\int_S q(\vec{r}\,')\frac{1}{4\pi\varepsilon|\vec{r}-\vec{r}\,'|}d\vec{r}\,'\bigg|_S = V_0$$

Electrostatics integral equation in 3D

Method of Moments (MoM) (2)

- The unknown charge is discretized as a linear combination of **basis functions**:

$$q_N(\vec{r}) = \sum_{n=1}^{N} q_n x_n(\vec{r})$$

$$\mathcal{L}q(\vec{r}) = V_0(\vec{r}) \qquad \mathcal{L}q_N(\vec{r}) = \sum_{n=1}^{N} q_n \mathcal{L}x_n(\vec{r}) \approx V_0(\vec{r})$$

  The **unknowns are now the coefficients** q_n

- But the equation is still a functional equation, and we need a linear system to solve with a computer.

- We want a very small (negligible) residual error:

$$R = V_0(\vec{r}) - \sum_{n=1}^{N} q_n \mathcal{L}x_n(\vec{r}) \approx 0$$

Method of Moments (MoM) (3)

- We set to zero inner products of the residual error

$$R(\vec{r}) = V_0(\vec{r}) - \sum_{n=1}^{N} q_n \mathcal{L} x_n(\vec{r})$$

with a set of **weighting functions** $w_m(\vec{r})$

where the inner product is: $\langle w(\vec{r}), f(\vec{r})\rangle = \int w^*(\vec{r}) f(\vec{r})\, d\vec{r}$

$$\langle w_m(\vec{r}), R(\vec{r})\rangle = \langle w_m(\vec{r}), V_0(\vec{r})\rangle - \sum_{n=1}^{N} q_n \langle w_m(\vec{r}), \mathcal{L} x_n(\vec{r})\rangle = 0$$

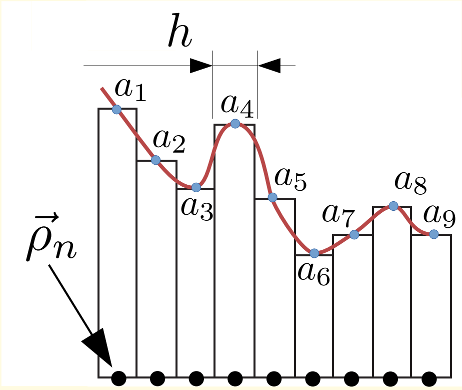
$$w_m(\vec{r}) = \delta(\vec{\rho} - \vec{\rho}_m) \Rightarrow R(\vec{r}_m) = V_0(\vec{r}_m) - \sum_{n=1}^{N} q_n \, \mathcal{L} x_n(\vec{r})\big|_{\vec{r}=\vec{r}_m} = 0$$

"Point matching"

- **Linear system**:

$$[Z][q] = [b] \qquad Z_{mn} = \mathcal{L} x_n(\vec{r})\big|_{\vec{r}=\vec{r}_m}, \quad b_m = V_0(\vec{r}_m)$$

Pulse basis functions and point matching:



Unit pulse basis functions

$$\frac{-1}{2\pi\varepsilon}\int_S q(\vec{\rho}\,')\ln\left(|\vec{\rho}-\vec{\rho}\,'|\right)d\vec{\rho}\,'\Big|_S = V_0$$

$$[Z][q]=[b] \qquad \begin{aligned} Z_{mn} &= \mathcal{L}x_n\left(\vec{\rho}\right)\big|_{\vec{\rho}=\vec{\rho}_m} \\ b_m &= V_0\left(\vec{\rho}_m\right) \end{aligned}$$

$$\mathcal{L}x_n(\vec{\rho}) = \frac{-1}{2\pi\varepsilon}\int_S \Pi\left(\frac{\vec{\rho}\,'-\vec{\rho}_n}{h_n}\right)\ln\left(|\vec{\rho}-\vec{\rho}\,'|\right)d\vec{\rho}\,'$$

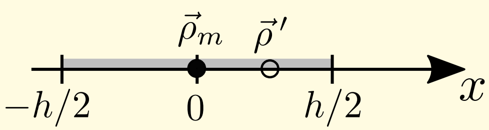
- With 1-point integration: $\mathcal{L}x_n(\vec{\rho}) \approx \dfrac{-1}{2\pi\varepsilon}h_n\ln(|\vec{\rho}-\vec{\rho}_n|)$
- Point matching at $\vec{\rho}_m$: $Z_{mn} = \dfrac{-h_n}{2\pi\varepsilon}\ln(|\vec{\rho}_m-\vec{\rho}_n|), \quad b_m = V_0(\vec{\rho}_m)$

MoM Electrostatics 2D (2)

$$Z_{mn} = \frac{-h_n}{2\pi\varepsilon}\ln(|\vec{\rho}_m - \vec{\rho}_n|), \quad b_m = V_0(\vec{\rho}_m)$$

- **Self-interaction** (diagonal terms): when $\vec{\rho}_m = \vec{\rho}_n$, $Z_{mm} = \infty$

  For $m = n$ we have to do the source integral analytically



$$Z_{mm} = \mathcal{L}x_m(\vec{\rho}_m) = \frac{-1}{2\pi\varepsilon}\int_S \Pi\left(\frac{\vec{\rho}\,' - \vec{\rho}_m}{h_m}\right)\ln\left(|\vec{\rho}_m - \vec{\rho}\,'|\right) d\vec{\rho}\,'$$

$$= \frac{-1}{2\pi\varepsilon}\int_{x_m}\ln\left(|\vec{\rho}_m - \vec{\rho}\,'|\right) d\vec{\rho}\,' = \frac{-1}{2\pi\varepsilon}\int_{-h_m/2}^{h_m/2}\ln|x|dx$$

$$= \frac{-1}{2\pi\varepsilon}2\int_0^{h_m/2}\ln x dx = \frac{-1}{\pi\varepsilon}\left[x(\ln x - 1)\right]_0^{h_m/2} = \boxed{\frac{-h_m}{2\pi\varepsilon}\left[\ln\left(\frac{h_m}{2}\right) - 1\right] = Z_{mm}}$$

Programming tips in 2D

Discretization of the geometry:

- Use complex numbers

$\vec{\rho}$: `r = x + j y;`

$\vec{\rho}_1 - \vec{\rho}_2$: `r1-r2;`

$|\vec{\rho}_1 - \vec{\rho}_2|$: `abs(r1-r2)`

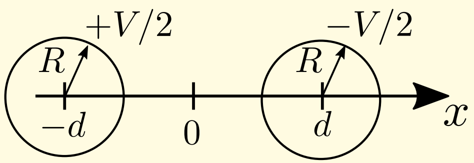
- Write a function that returns a vector `rn` with the center point of the basis functions and another vector `hn` with the lengths

2D plot:

- Use meshgrid:

```
x = linspace(...), y = linspace(...)
[xx,yy] = meshgrid(x,y); rr = xx + 1j*yy;
```

- Loop BF: `V=zeros(...); for n=1:N, R = abs(rr-rn(n)); V=V+...; end`
- Potential and field: use `surf(),pcolor(),contour(),gradient(),quiver()`



Error check:

- Exact capacitance of two infinite cylinders (per u.L.):

$$C = \frac{\pi\varepsilon}{\operatorname{arccosh}\left(\frac{d}{R}\right)}$$

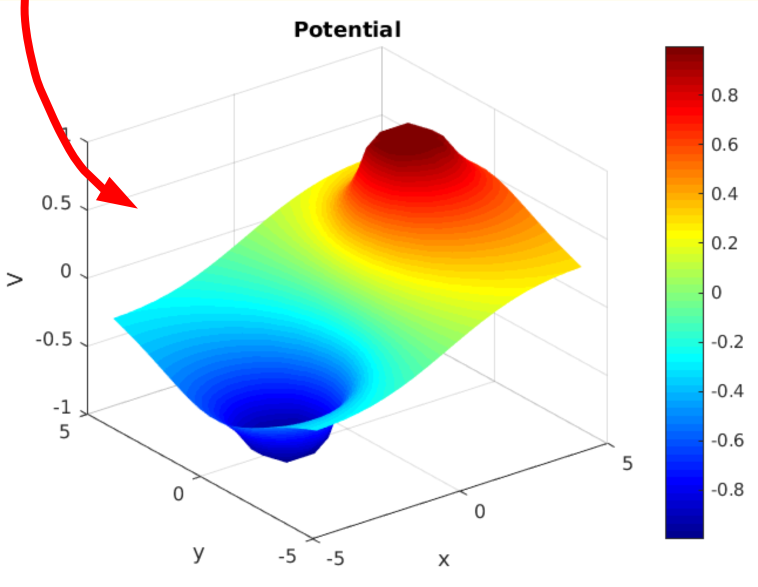
- Computation

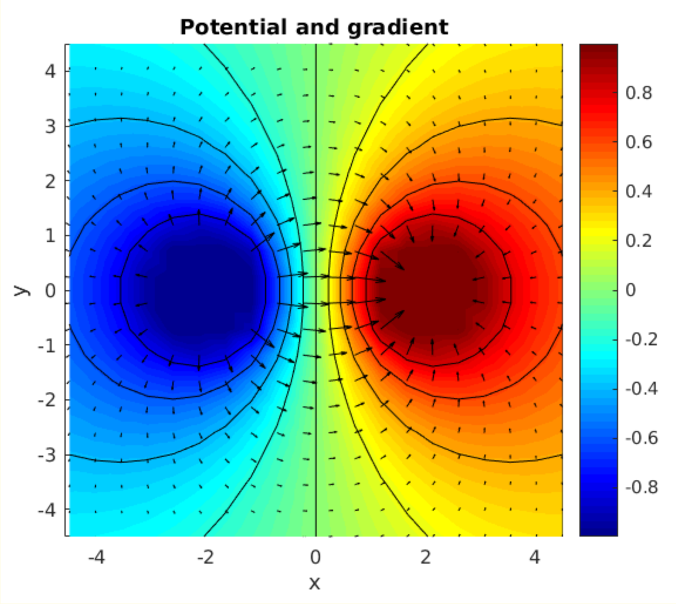
$$C = \frac{Q}{V_0} = \frac{1}{V_0}\sum_{n=1}^{N} h_n q_n = \texttt{sum(hn.*qn)/V}$$

Programming tips in 2D (2)

- **Post-processing:**
  - **2D plots:**

```
x = linspace(...), y = linspace(...);
[xx,yy] = meshgrid(x,y); rr = xx + 1j*yy;
for …, V = … ; end; % Computation of potential at xx,yy
[Ex, Ey] = gradient(-V);
surf(x,y,V);
pcolor(x,y,V); quiver(xx,yy,Ex,Ey); contour(xx,yy,V):
colormap jet; shading interp; colorbar
```





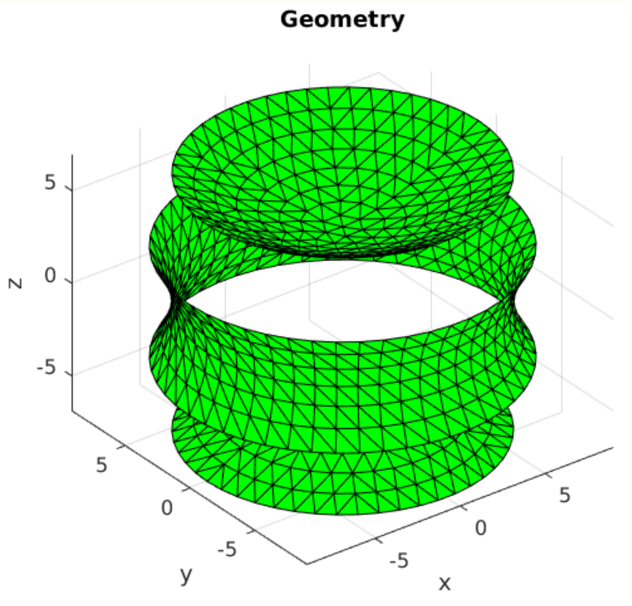
Work to do (in 2D)

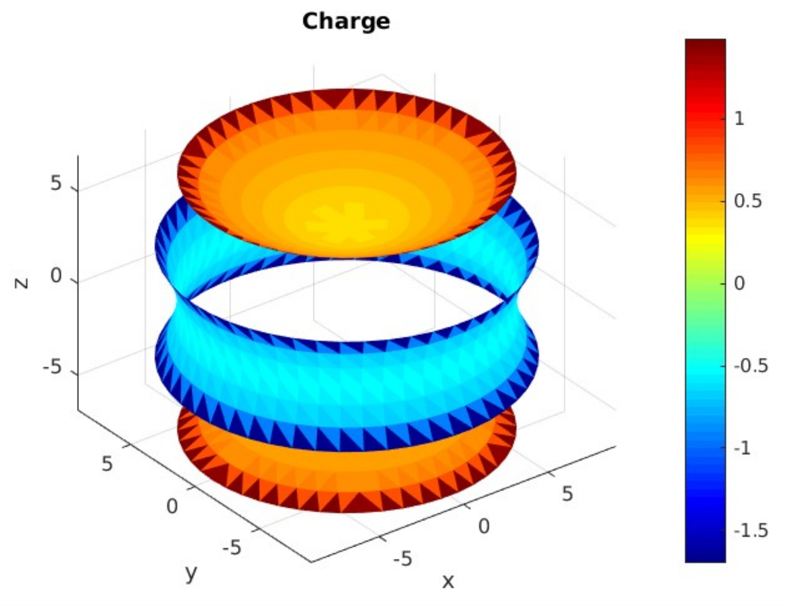
- **Object geometry:**
  - Create functions that return `rn` and `hn` for planar and circular objects.
  - Create an object consisting of two (planar or circular) plates.
  - Plot the geometry.
- **Linear System:**
  - Compute linear system elements.
  - Compute independent term: set one plate to +V/2 and the other to -V/2.
  - Solve linear system and compute capacitance. Check with reference.
  - Compute potential and field.
- **Plot potential and E field:**
  - Obtain `surf(), pcolor(), contour()` and `quiver()` plots of potential and field.

MoM Electrostatics 3D

$$\int_S q(\vec{r}\,')\frac{1}{4\pi\varepsilon|\vec{r}-\vec{r}\,'|}d\vec{r}\,'\Bigg|_S = V_0$$

- **Triangular basis functions and point matching:**
  - Planar triangles mesh
  - Constant charge on each triangle



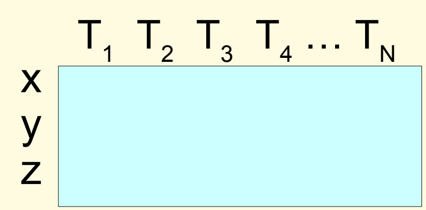


MoM Electrostatics 3D (2)

- **MoM matrix elements:**
  - The integral of $\mathcal{L}x_n(\vec{r})$ can be computed analytically

    (see Rao et al., “A simple numerical solution procedure for statics problems involving arbitrarily shaped surfaces” IEEE Trans AP, Vol. 27, No. 5, sept 1979)
  - Matlab function by E. Úbeda
    Computation of integral and point matching
    $$\int_{S_{T_n}} \frac{1}{|\vec{r}_m - \vec{r}\,'|} d\vec{r}\,'$$
    Returns **a row** of [Z]: single observation point and all source triangles.

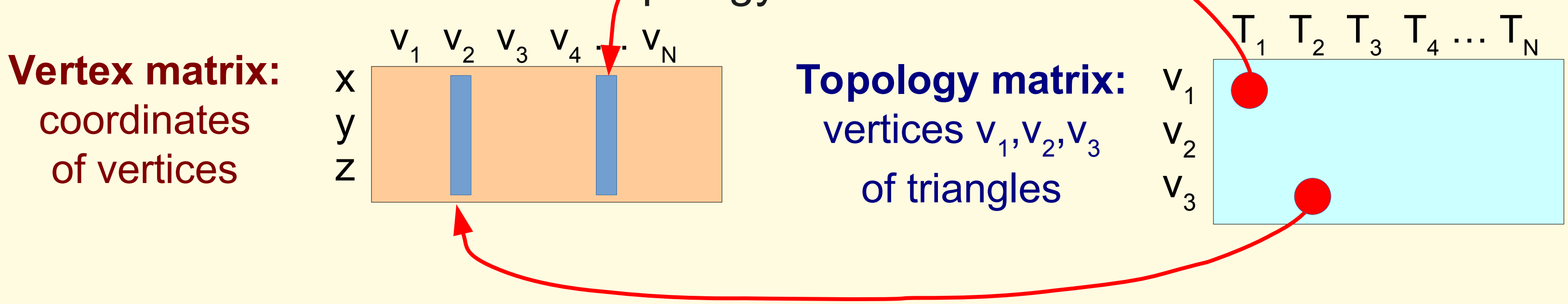
```
int_S_1divR(rf, r_1, r_2, r_3, un_S, cent_S);
Output: intS
Input: rf: field point [3x1]
       r_1 : first vertex of triangles [3xNt]
       r_2 : second vertex of triangles [3xNt]
       r_3 : third vertex of triangles [3xNt]
       un_S: normal vectors to the set of triangles <r1-r2-r3> [3xNt]
       < IMPORTANT: r1-r2-r3 must turn in the same sense as un_S >
       cent_S: set of centroids of the triangles <r1-r2-r3> [3xNt]
```



Programming tips in 3D

Discretization of the geometry

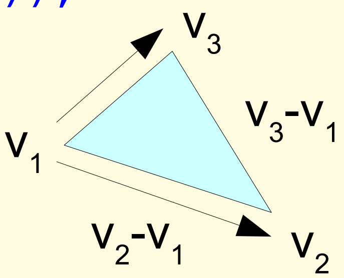
- Use a vertex matrix and a topology matrix



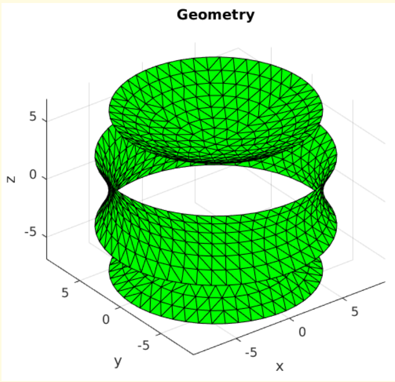
- Write a function that returns a structure with 'vertex' and 'topol' fields: obj.vertex and obj.topol = vertex and topology matrices
  - Create a cloud of surface points, possibly with `x = linspace(...), y = linspace(...), [xx,yy] = meshgrid(x,y), zz = fun(xx,yy)`
  - `obj.vertex = [xx(:) yy(:) zz(:)]';`
  - Use `obj.topol = delaunay(xx(:), yy(:))';` Matlab function
- We also provide a function that reads a FEM triangular mesh from a file created with GiD mesher (www.gidhome.com) and returns the obj struct

Programming tips in 3D (2)

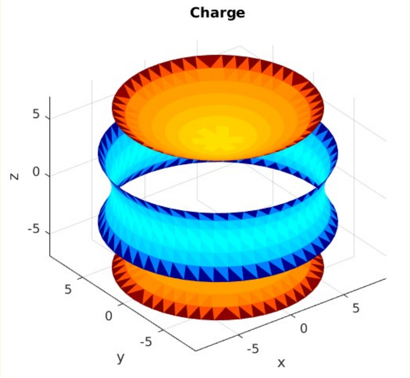
- **Geometry management:** `int_S_1divR()` **function parameters**
  - **v1, v2, v3 of all triangles:**
```
v1 = obj.vertex(:,obj.topol(1,:));
v2 = obj.vertex(:,obj.topol(2,:));
v3 = obj.vertex(:,obj.topol(3,:));
```
  - **Centroid of all triangles:**
```
obj.cent =(v1+v2+v3)/3;
```



  - **Unit normal and area of triangles:**
```
c  = cross(v2-v1, v3-v1);
obj.ds = sqrt(sum(c.^2))/2;
obj.un = c./repmat(2*obj.ds,3,1); % Normalization
```



  - **View geometry:**
```
patch('Faces',obj.topol','Vertices',obj.vertex',
'FaceColor','g','EdgeColor','k');
axis equal;
```



  - **With charge:**
```
patch('Faces',obj.topol','Vertices',obj.vertex',
'CData',q,'FaceColor','flat','EdgeColor','none');
axis equal;
```

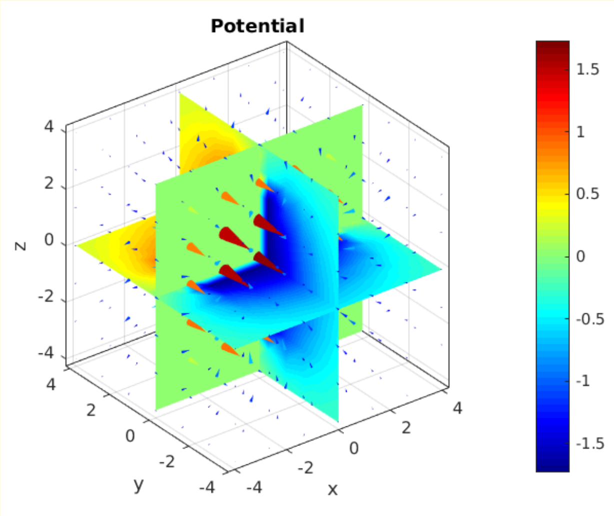
Programming tips in 3D (3)

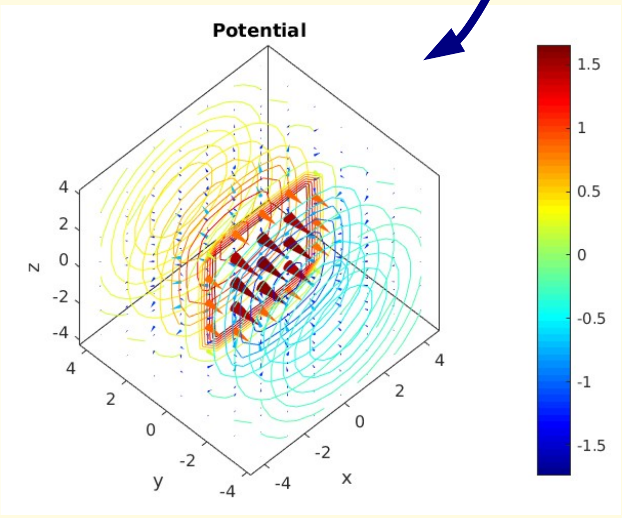
- **Post-processing:**
  - **3D plots:**

```
x = linspace(...), y = linspace(...); z = linspace(...);
[xx,yy,zz] = meshgrid(x,y,z);

for …, V = … ; end; % Computation of potential at xx,yy,zz

[Ex, Ey, Ez] = gradient(-V);
slice(xx,yy,zz, V, xs,ys,zs);
contourslice(xx,yy,zz, V, xs,ys,zs);
coneplot(xx,yy,zz, Ex,Ey,Ez, cxx,cyy,czz, absE);
colormap jet; shading interp; colorbar
```



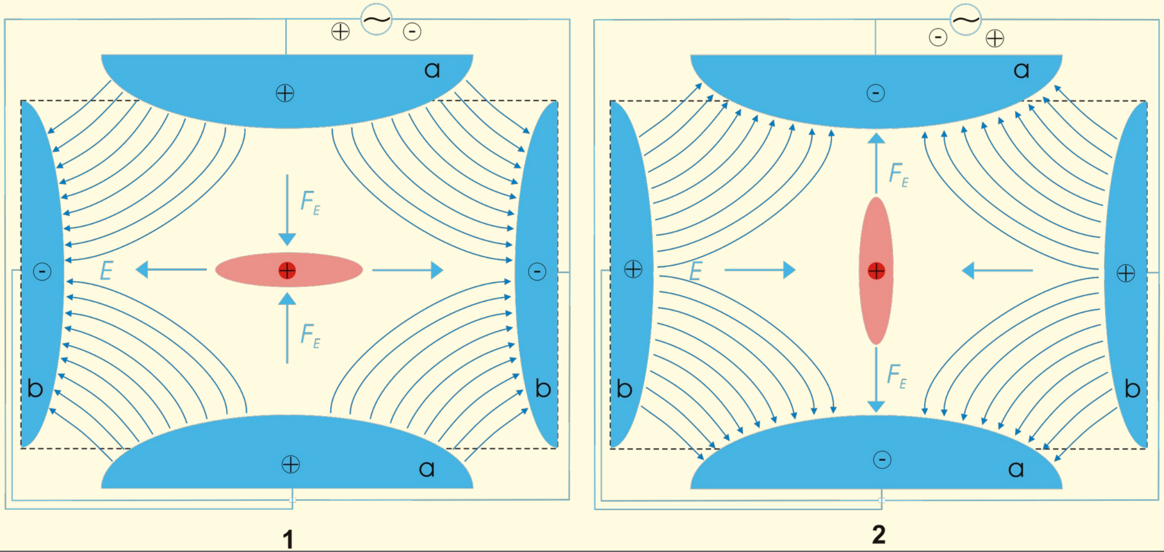


Work to do (in 3D)

- **Object geometry:**
  - Create function that returns `obj.vertex` and `obj.topol` for planar plate.
  - Create an object consisting of two planar plates. Plot the geometry.
- **Linear System:**
  - Compute the arguments of `int_S_1divR()` function.
  - Compute the linear system elements.
  - Compute independent term: set one plate to +V/2 and the other to -V/2.
  - Solve linear system and compute capacitance. Check with reference.
  - Compute potential and field.
- **Plot potential and E field:**
  - Obtain `slice()`, `contourslice()` and `coneplot()` plots of potential and field.

Final work

- **Methodology:**
  - Develop computer code for computing the potential due to hyperbolic quadrupole
  - Compute ion trajectories within the trap.
  - Account for AC variation of potential → ion trajectories
  - Design size of the trap, AC frequency, etc… for ion confinement with weights and charge.



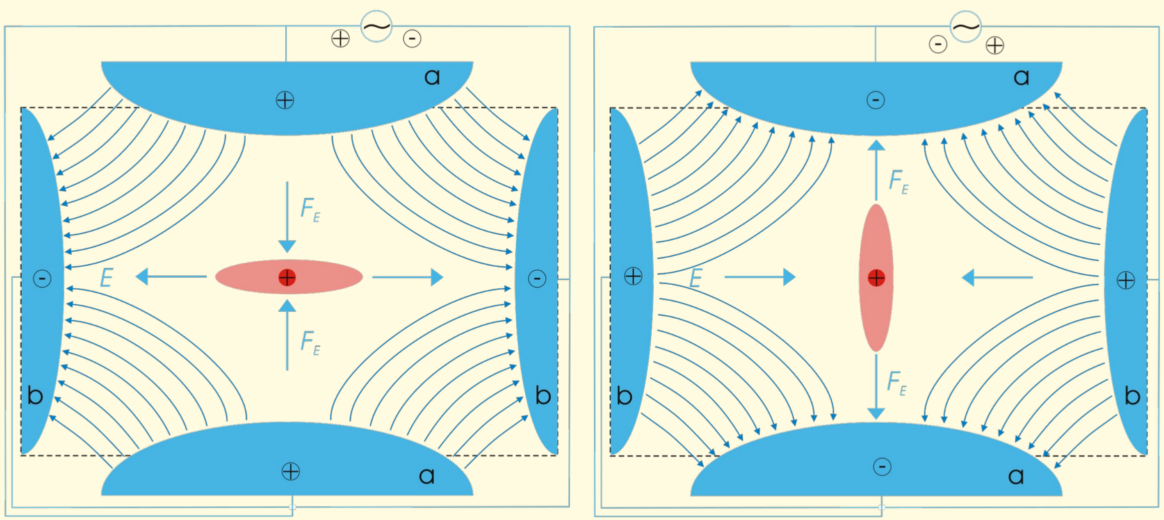
Final work (2)

Methodology (cont):

With only one ion in the trap (initially static), we just have to lower the voltage to ensure that it does not escape. For $V_0=0$ it just remains static.

Set **two (or more) equal ions** in the trap, at random positions: repulsion will make them escape if V_0 is too low.

- Compute the two ion trajectories **accounting for the repulsion force** between them. Extend to more than two ions, if possible.
- Determine the range for V_0 and AC frequency that confine the ions.



Schedule

- Week 1: Implementation in 2D: validation with capacitor
- Week 2: Implementation in 3D: validation with capacitor
- Week 3: Paul Trap: Hiperbolic electrodes, static potential
- Week 4: Trajectory of (i) one ion and (ii) two ions with repulsion
- Week 5: AC potential and trajectories of multiple ions
- Week 6: Design the trap for optimum ion confinement
  - Size of the trap, voltage V_0, AC frequency, etc...