LinkedIn

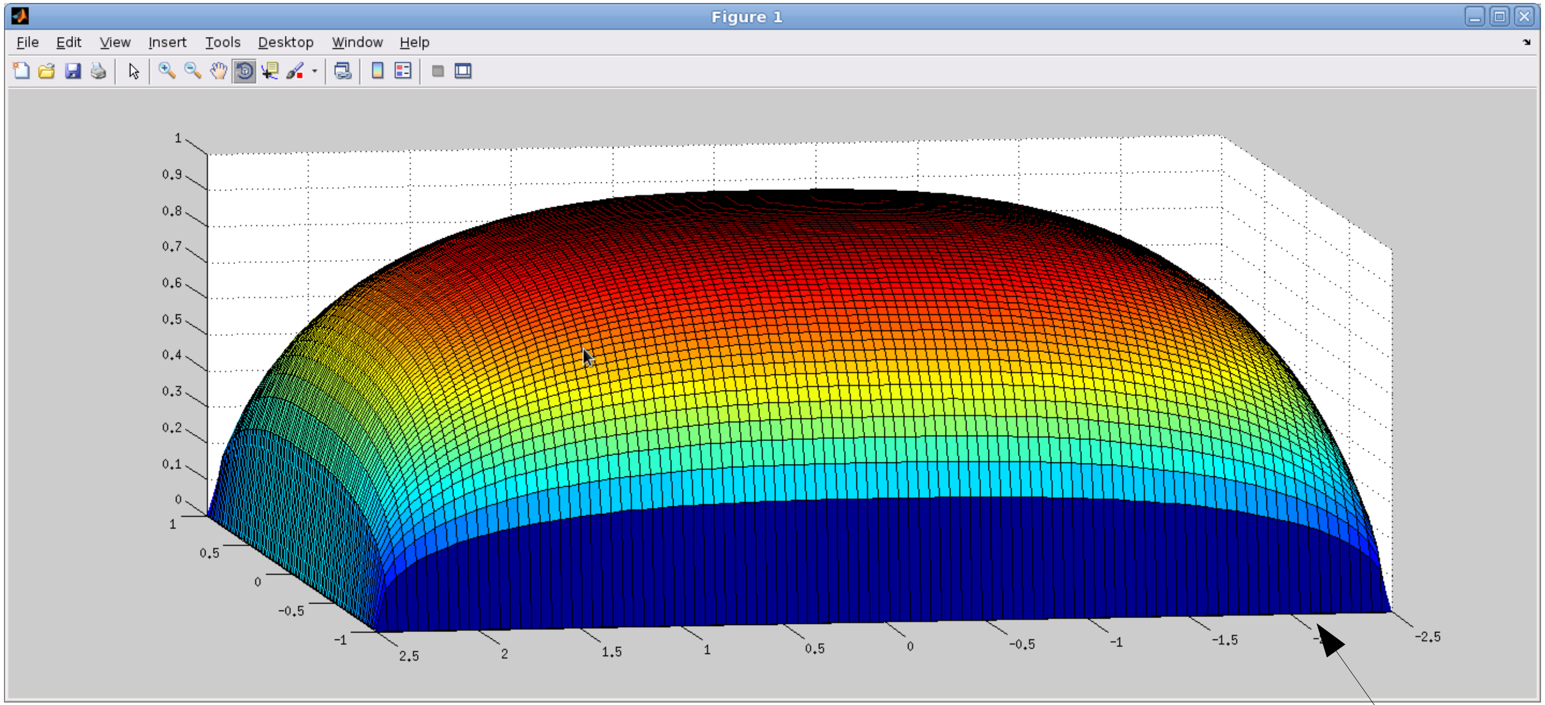
Integration in 2 dimensions

- Midpoint method in 2D
- Gaussian quadrature in 2D
- Extension to ND should be obvious after we think about 2D.

Example: Volume of tennis court

- Find volume of inflatable building.
- Useful computation for planning air conditioning (for example).

Compute volume using midpoint rule



- Model: $$z(x,y)=\left(1-\left|\frac{x}{a}\right|^{p}\right)^{1/p}\left(1-\left|\frac{y}{b}\right|^{p}\right)^{1/p}$$

p = 2.7

- Shape of dome depends upon parameter p.
- No closed form expression for integral?

Aside on superellipse

- Superellipse

$$\left(\left|\frac{x}{a}\right|^p + \left|\frac{y}{b}\right|^p\right)^{1/p} = c$$

- Parameter p "tunes" the shape of the superellipse.

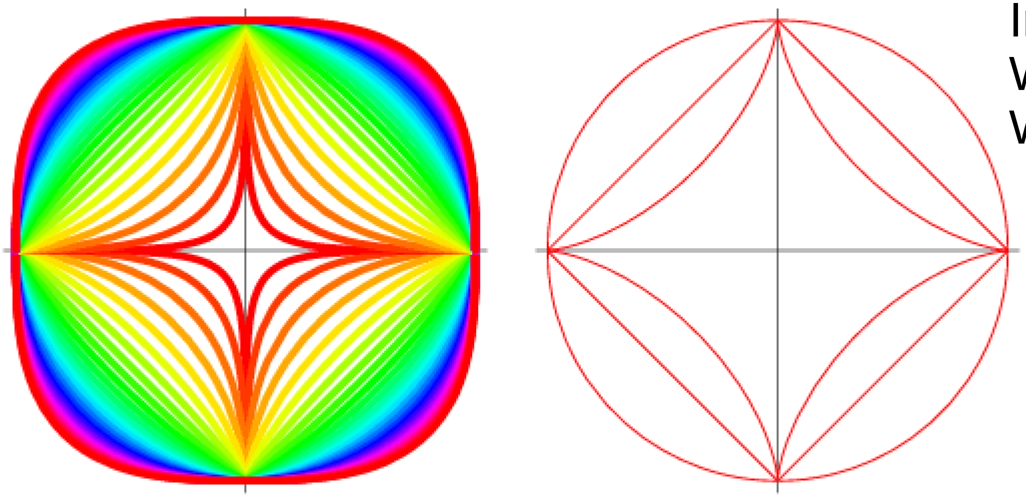


Image source: Wolfram Math World

- I model tennis court as superellipse in x and y directions.

P = 1

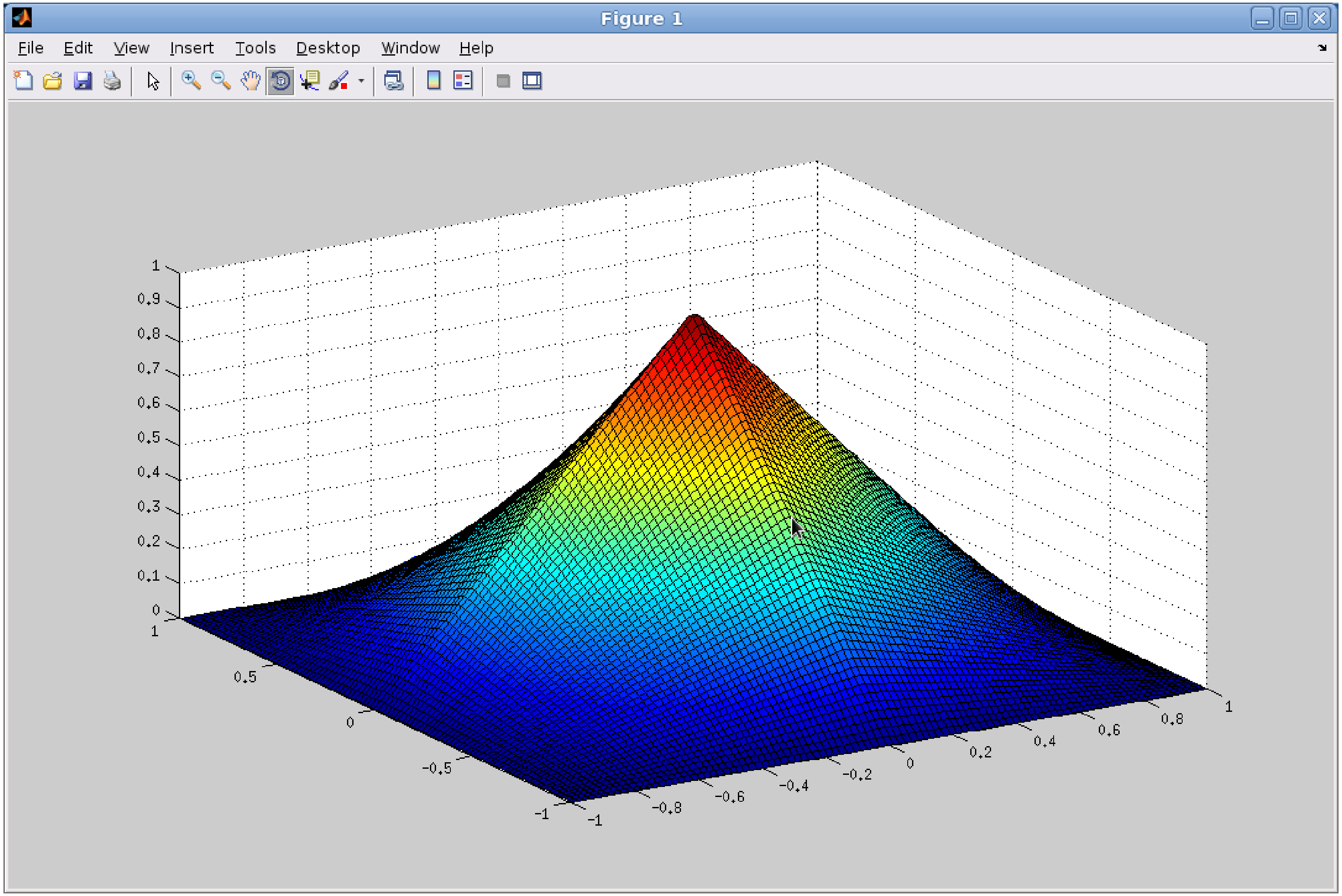


Figure 1
File Edit View Insert Tools Desktop Window Help
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
0.5
0
-0.5
-1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

P = 2

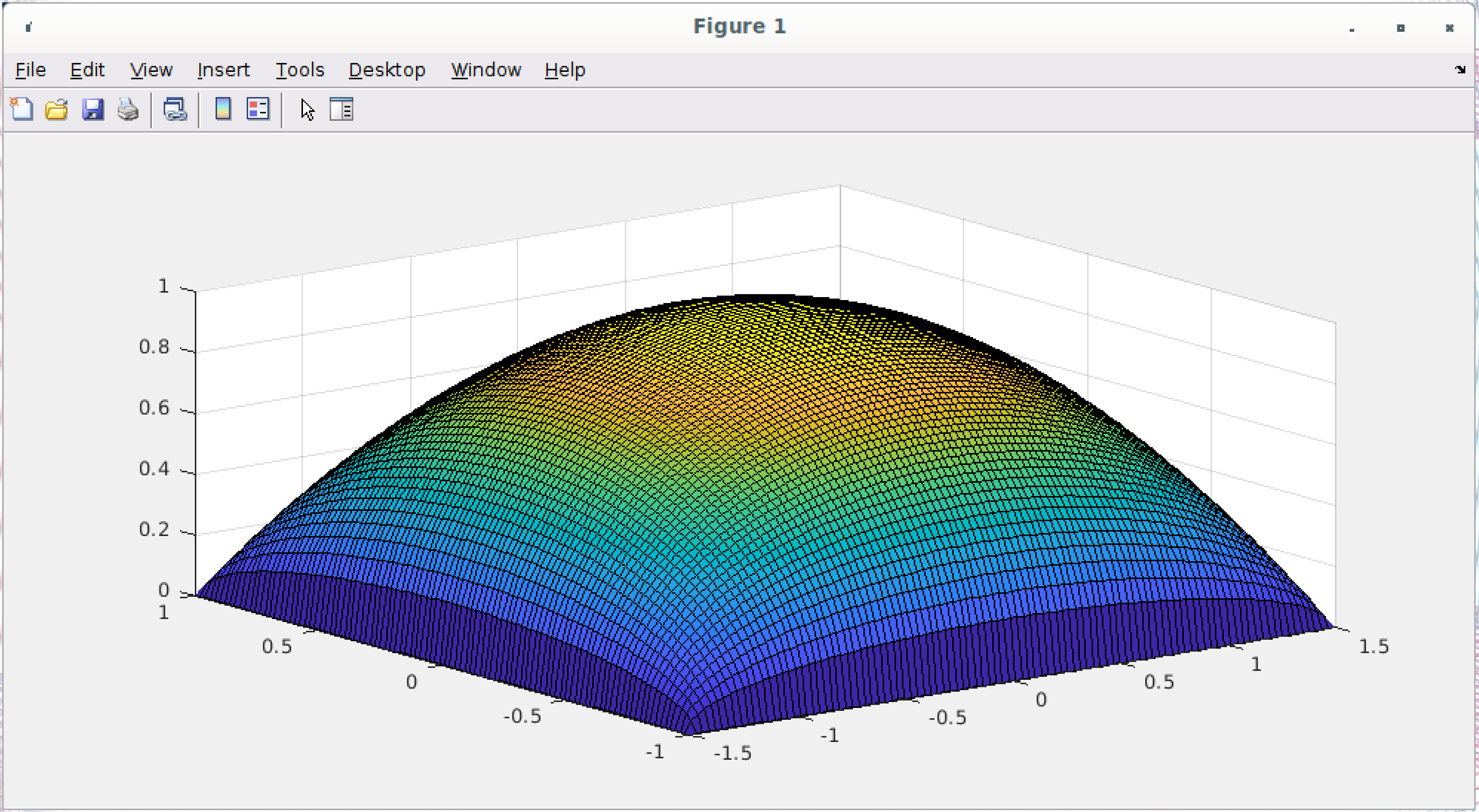


Figure 1
File Edit View Insert Tools Desktop Window Help
1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1.5
-1
-0.5
0
0.5
1
1.5

P = 2.7

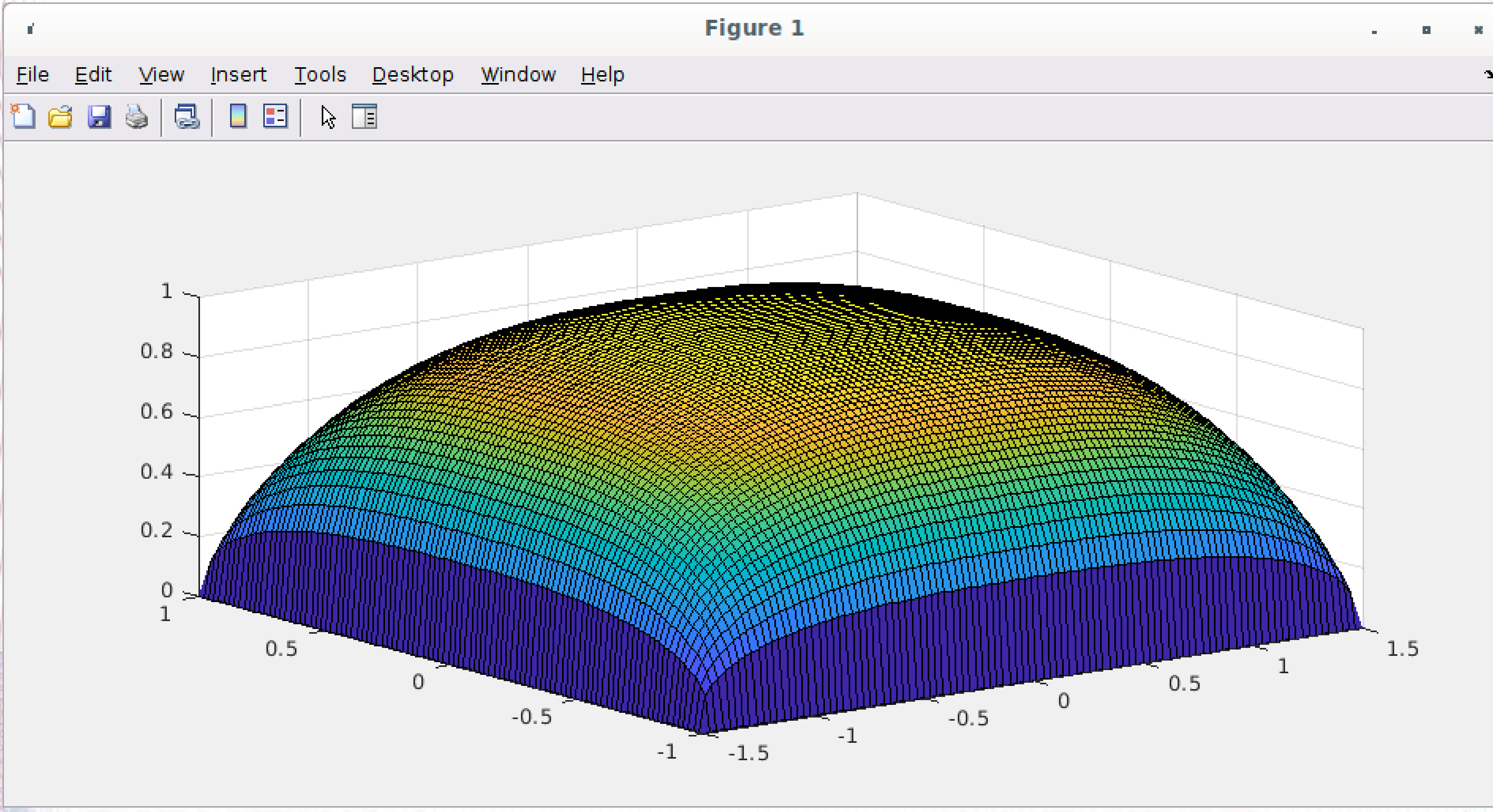
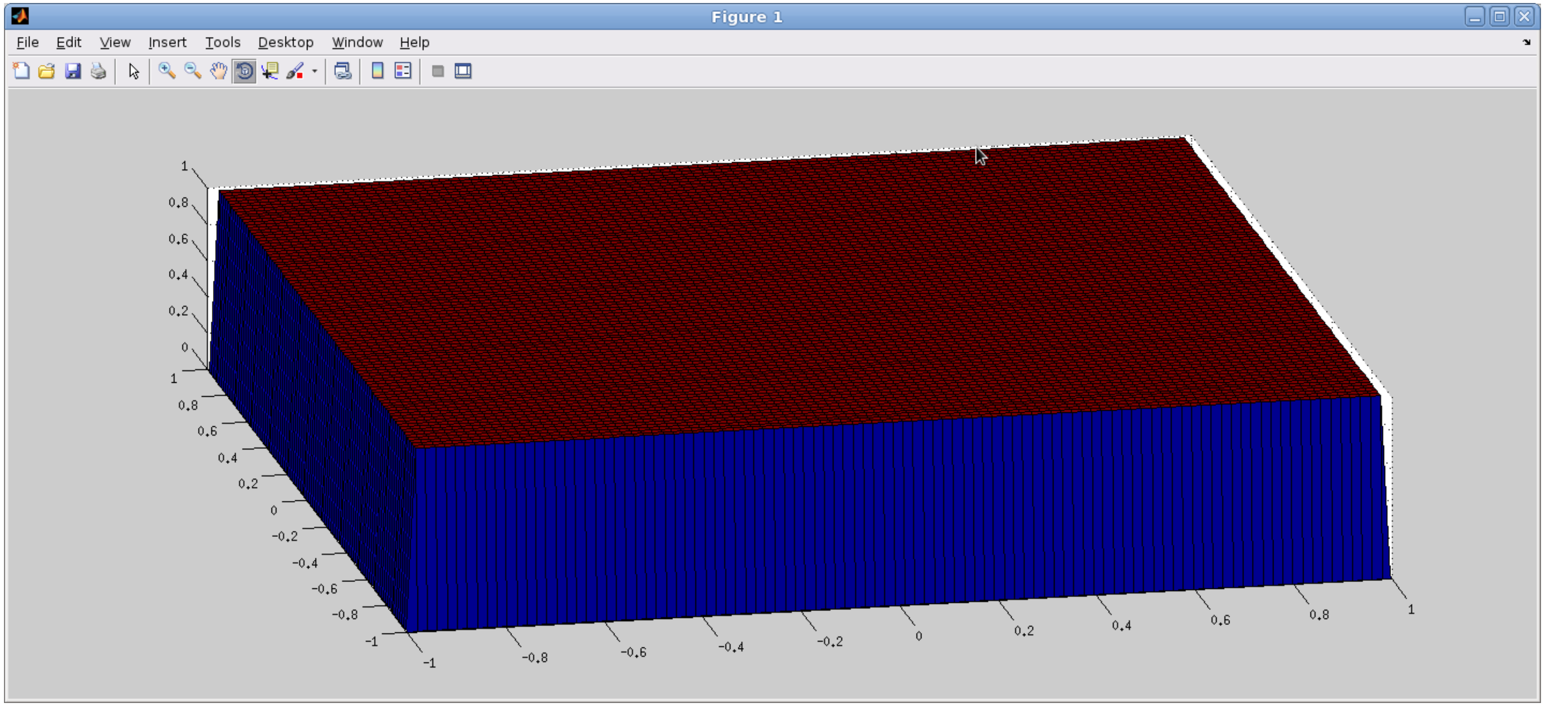


Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1.5
-1
-0.5
0
0.5
1
1.5

P = 1000

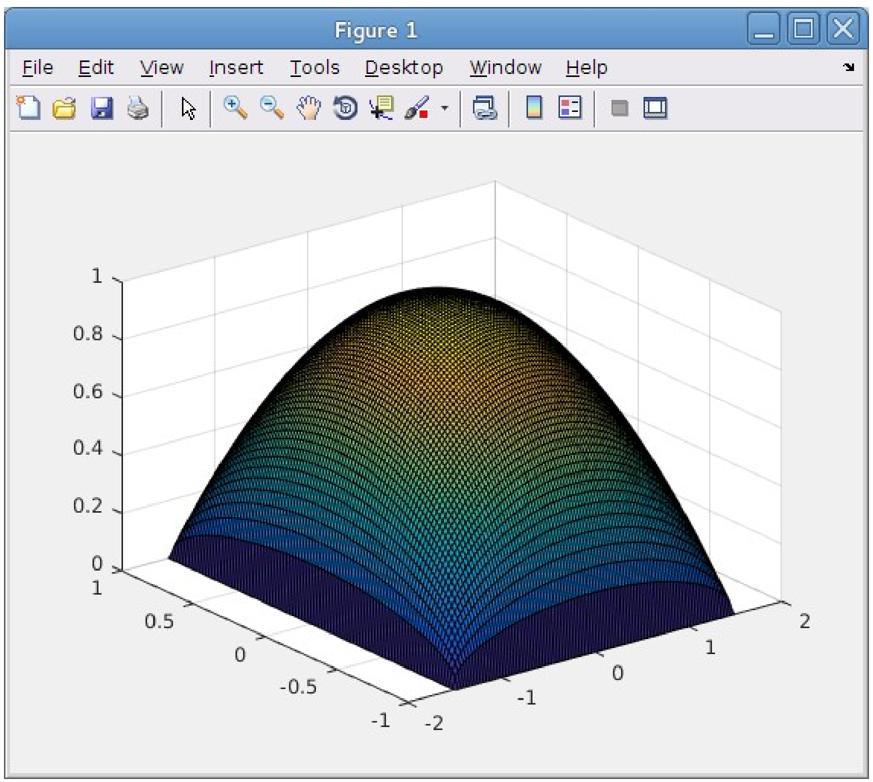


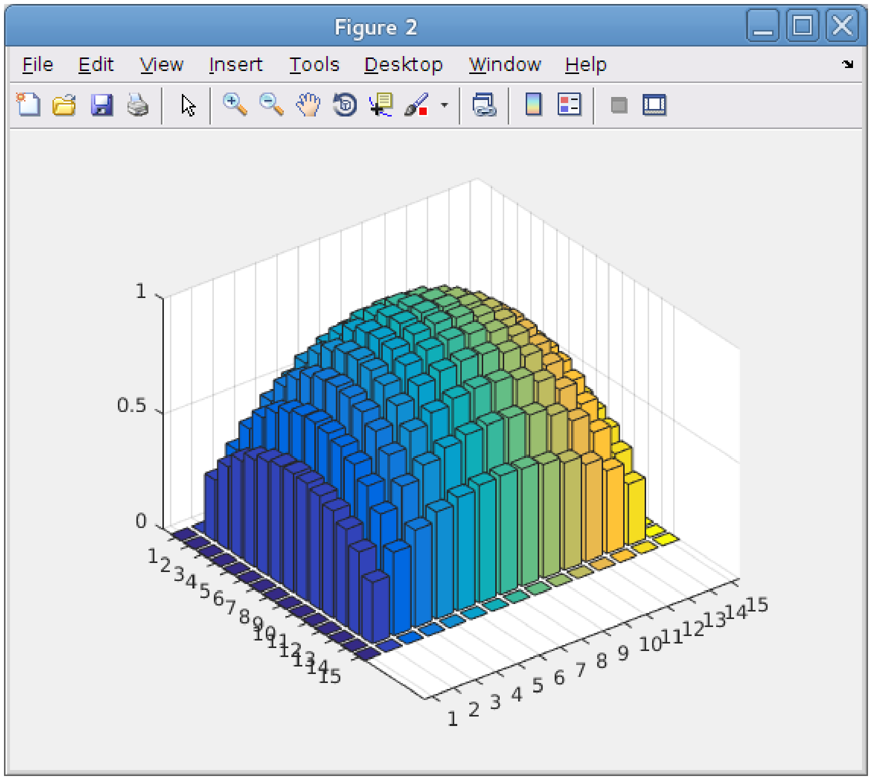
- Shape is a cuboid (rectangular cube)
- Good test case – we can easily compute the volume

Simplest idea: 2D midpoint method

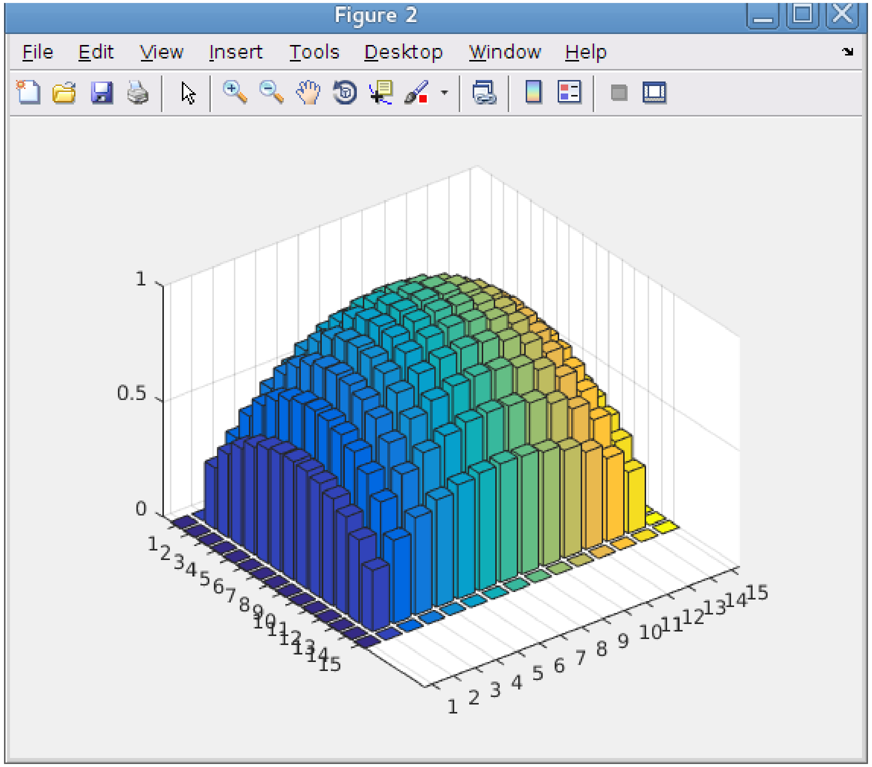
1. Divide region of interest up into rectangles (cells).

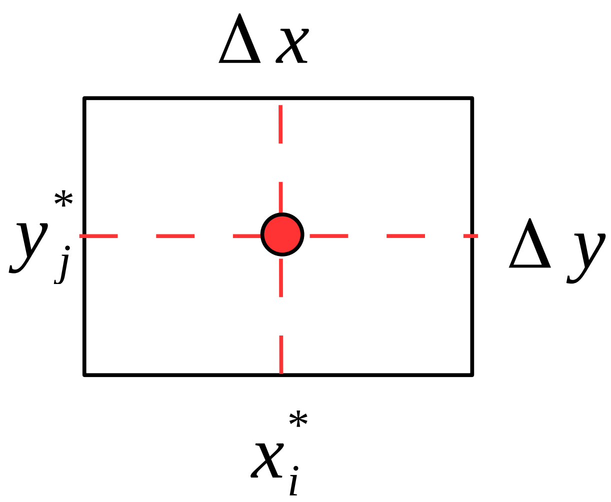
2. Compute integral using sum of function values at midpoints of cells.





2D midpoint method





$$I=\iint_A dx\,dy\,f(x,y)\approx\sum_{i=1}^{N_x}\sum_{j=1}^{N_y} f(x_i^*,y_j^*)\,\Delta x\,\Delta y$$

Star means evaluate the function at the mid point of the box.

```
% Chop up dome into smaller areas (cells)
x = linspace(-Lx/2, Lx/2, Nx);
dx = x(2) - x(1);
y = linspace(-Ly/2, Ly/2, Ny);
dy = y(2) - y(1);


% Now do midpoint-rule integration on each area
V = 0;            % Volume to compute
% Loop over sample points.  Don't use last one because
% it is at an edge
for xidx=1:(Nx-1)
  for yidx=1:(Ny-1)
    % Find point in middle of rectangle
    x0 = x(xidx) + dx/2;
    y0 = y(yidx) + dy/2;
    % Add value of fcn at this point to the running volume sum.
    ss = s(dome, x0, y0);
    V = V + ss*dx*dy;
  end
end
```

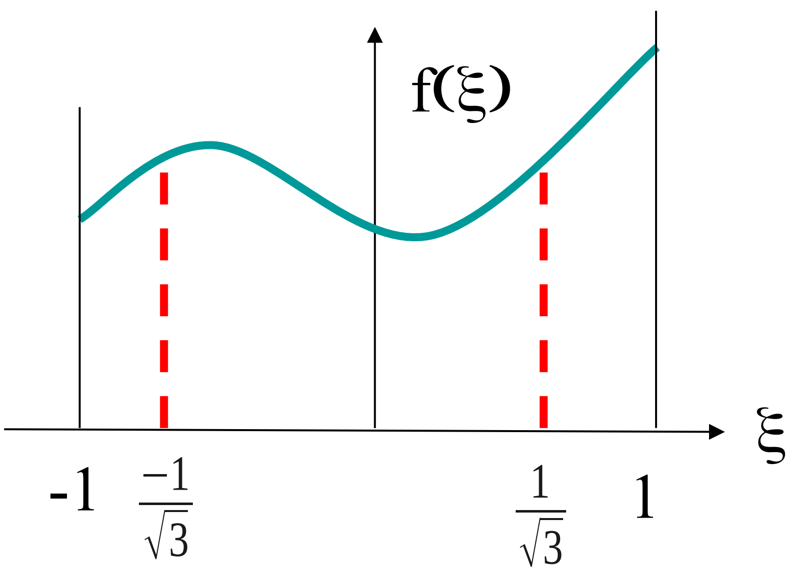
Midpoint rule in 2D

- Advantages:
  - Simple
- Disadvantages:
  - Not particularly accurate
  - Can't deal with non-rectangular domains.

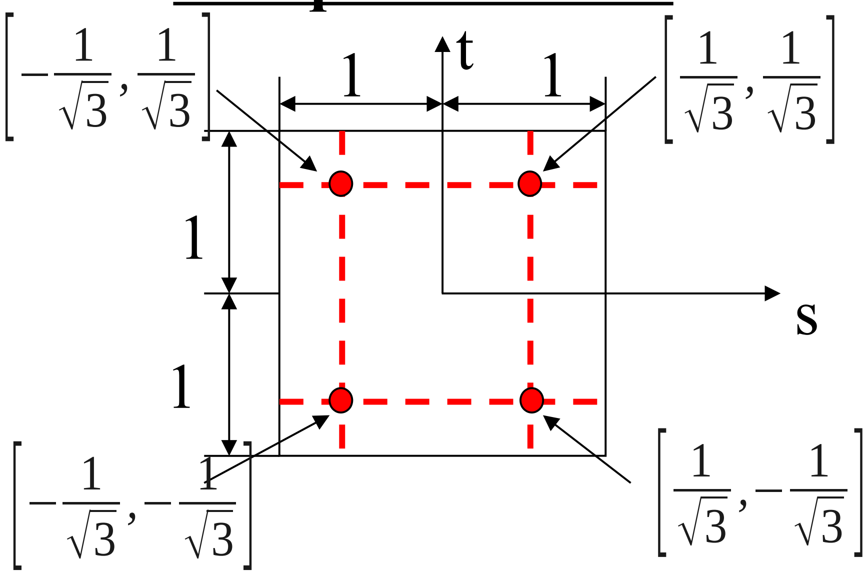
Can we use Gaussian quadrature directly in 2D?

- Yes – combine mid-point method with 2D interpolants for f(x, y).

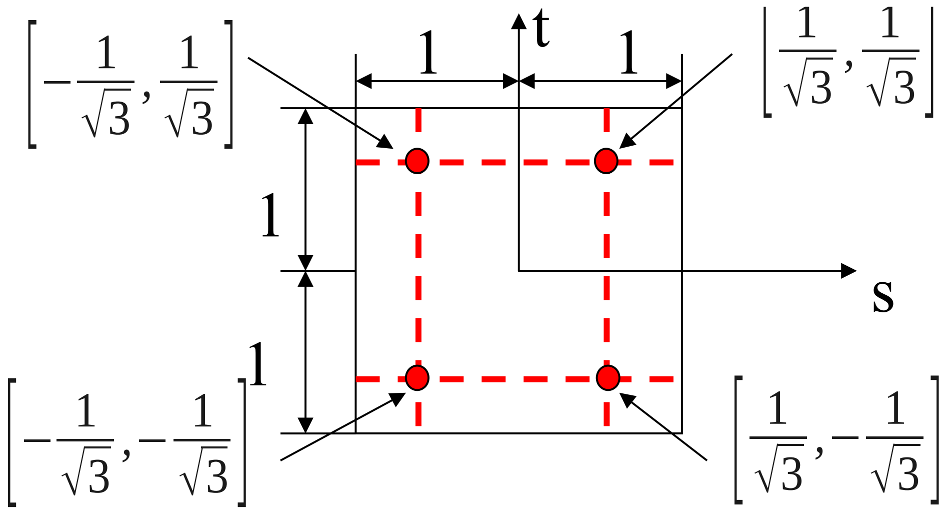
1D linear domain



2D square domain



2D Gaussian Quadrature M = 2



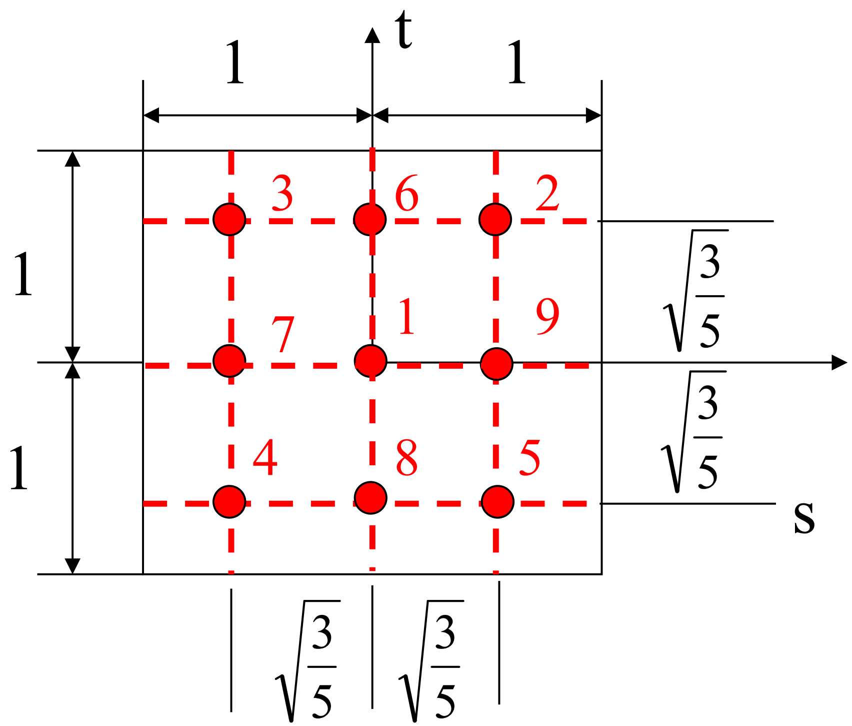
$$I = \int_{-1}^{1} ds \int_{-1}^{1} dt\, f(s,t) \approx \sum_{i=1}^{2} \sum_{j=1}^{2} w_{ij} f(s_i, t_j)$$

$$= f(r,r) + f(r,-r) + f(-r,r) + f(-r,-r)$$

with $r = \frac{1}{\sqrt{3}}$ and $w_{ij}^{(2)} = w_i^{(1)} w_j^{(1)} = 1$

- 2D weights are products of 1D weights
- Formula is exact for product of two cubics

2D Gaussian Quadrature M = 3



$$w^{(2)}(1)=w_i^{(1)}w_j^{(1)}=\frac{8}{9}\cdot\frac{8}{9}=\frac{64}{81}$$

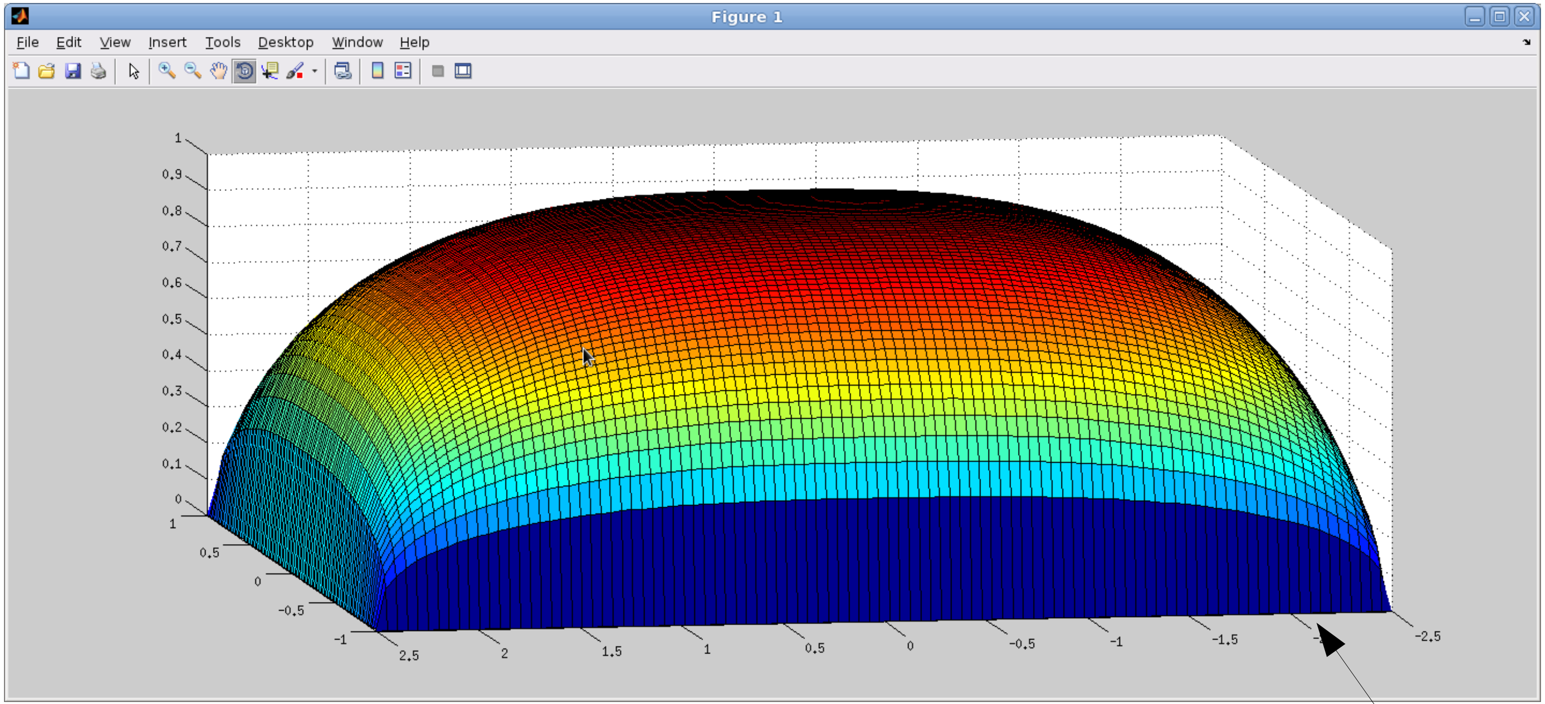
$$w^{(2)}(2,3,4,5)=\frac{5}{9}\cdot\frac{5}{9}=\frac{25}{81}$$

$$w^{(2)}(6,7,8,9)=\frac{5}{9}\cdot\frac{8}{9}=\frac{40}{81}$$

$$I=\int_{-1}^{1} ds \int_{-1}^{1} dt\, f(s,t)\approx\sum_{i=1}^{3}\sum_{j=1}^{3} w_{ij} f(s_i,t_j)$$

- 2D weights are products of 1D weights
- Method is exact for product of two 5th degree polynomials

Compute volume using Gaussian quadrature



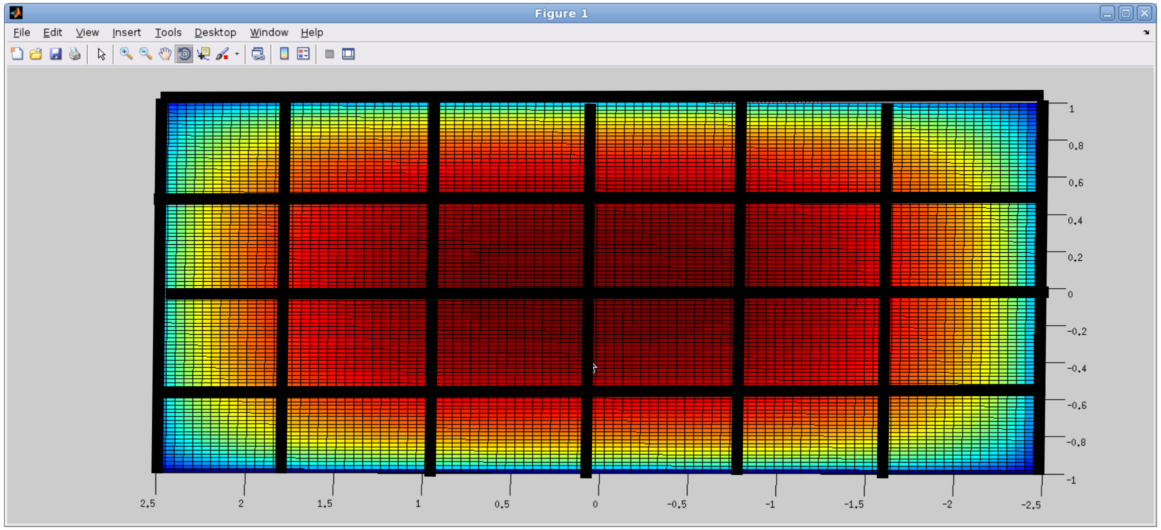
p = 2.7

- Model: $$z(x,y) = \left(1 - \left|\frac{x}{a}\right|^{p}\right)^{1/p} \left(1 - \left|\frac{y}{b}\right|^{p}\right)^{1/p}$$
- Shape of dome depends upon parameter p.
- No closed form expression for integral?

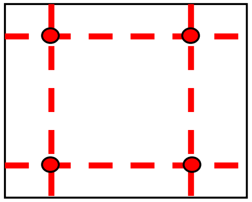
Compute integral using M=2 Gaussian Quadrature

$$V=\int_{-L_x/2}^{L_x/2}dx\int_{-L_y/2}^{L_y/2}dy\left(1-\left(\frac{x}{a}\right)^p\right)^{1/p}\left(1-\left(\frac{y}{b}\right)^p\right)^{1/p}$$

- Chop domain into boxes:



- Do M=2 Gaussian quadrature on each box:



```
% Chop up dome into smaller areas
x = linspace(-Lx/2, Lx/2, Nx);
dx = x(2) - x(1);
y = linspace(-Ly/2, Ly/2, Ny);
dy = y(2) - y(1);

% Now do Gaussian quadrature integration on each area
V = 0;            % Volume to compute
hx = dx/(2*sqrt(3));  % Define convenience variable.
hy = dy/(2*sqrt(3));  % Define convenience variable.
% Loop over sample points.  Don't use last one because
% it is at an edge
for xidx=1:(Nx-1)
  for yidx=1:(Ny-1)
    % Find point in middle of rectangle
    x0 = x(xidx) + dx/2;
    y0 = y(yidx) + dy/2;

    % Now compute value of fcn at quadrature points
    spp = s(dome, x0 + hx, y0 + hy);
    spm = s(dome, x0 + hx, y0 - hy);
    smp = s(dome, x0 - hx, y0 + hy);
    smm = s(dome, x0 - hx, y0 - hy);
    V = V + (spp+spm+smp+smm)*dx*dy/4;
  end
end
```

Test integration for different p

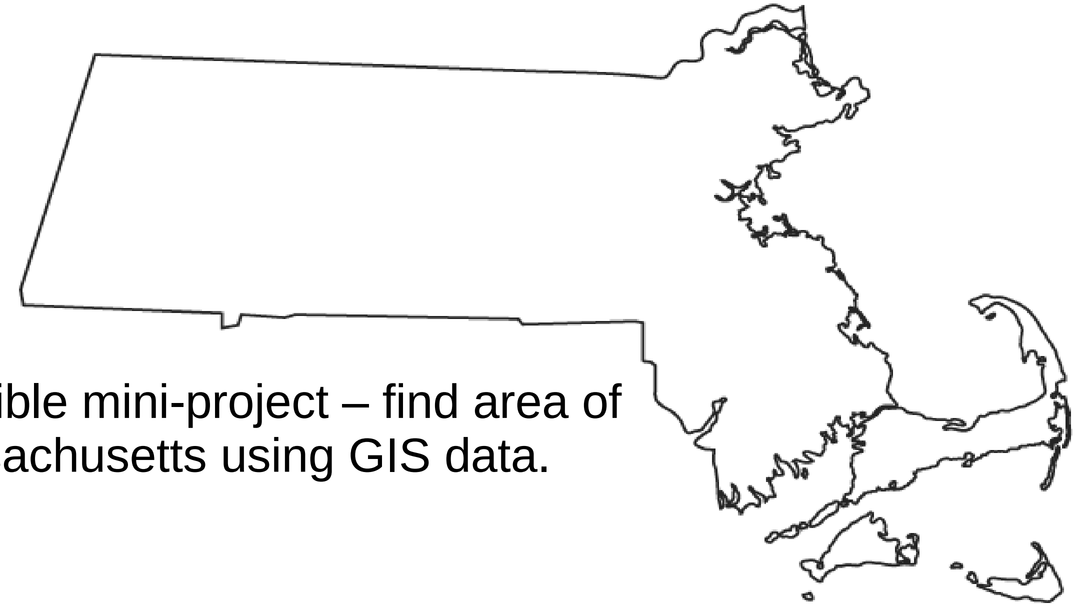
```
Integrating dome with p=1.  Answer should be 1.500000
Testing accuracy of 2D Gaussian quadrature
p = 1.000000, Vgauss = 1.500000, Vtrue = 1.500000, diff = 4.440892e-16 ...  Passed!
Testing accuracy of 2D midpoint rule
p = 1.000000, Vmid = 1.500000, Vtrue = 1.500000, diff = 2.220446e-16 ...  Passed!
------------------------------------------------------------------------------------
Integrating dome with p=2.  Answer should be 3.701102
Testing accuracy of 2D Gaussian quadrature
p = 2.000000, Vgauss = 3.701735, Vtrue = 3.701102, diff = -6.328837e-04 ...  Passed!
Testing accuracy of 2D midpoint rule
p = 2.000000, Vmid = 3.855406, Vtrue = 3.701102, diff = -1.543042e-01 ...  Failed
------------------------------------------------------------------------------------
Integrating dome with p=4.  Answer should be 5.156389
Testing accuracy of 2D Gaussian quadrature
p = 4.000000, Vgauss = 5.158872, Vtrue = 5.156389, diff = -2.482626e-03 ...  Failed
Testing accuracy of 2D midpoint rule
p = 4.000000, Vmid = 5.347053, Vtrue = 5.156389, diff = -1.906632e-01 ...  Failed
------------------------------------------------------------------------------------
Integrating dome with p=10000.  Answer should be 6.000000
Testing accuracy of 2D Gaussian quadrature
p = 10000.000000, Vgauss = 6.000000, Vtrue = 6.000000, diff = -1.776357e-15 ...
Passed!
Testing accuracy of 2D midpoint rule
p = 10000.000000, Vmid = 6.000000, Vtrue = 6.000000, diff = -1.776357e-15 ...  Passed!
------------------------------------------------------------------------------------
Using Gaussian quadrature, with p = 2.700000, tennis court volume = 4.463983
Using 2D midpoint rule, with p = 2.700000, tennis court volume = 4.652684
```

Remarks

- Accuracy will increase with:
  - Increased quadrature order (M)
  - Increased number of integration cells
- Extension to higher dimensions obvious.
- We still have not solved the problem of integration over irregular domains.

Integrating over odd-shaped boundaries

- Example: Find area of Massachusetts



Possible mini-project – find area of Massachusetts using GIS data.

A note on integration

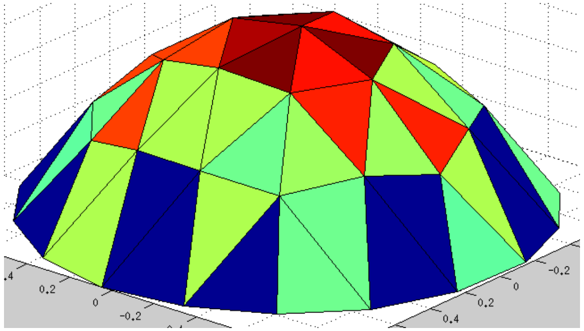
- Find surface area of a domain

$$\iint_{(x,y)\in A} dx\,dy\,1$$

Possible mini-project – find area of Massachusetts using GIS data.

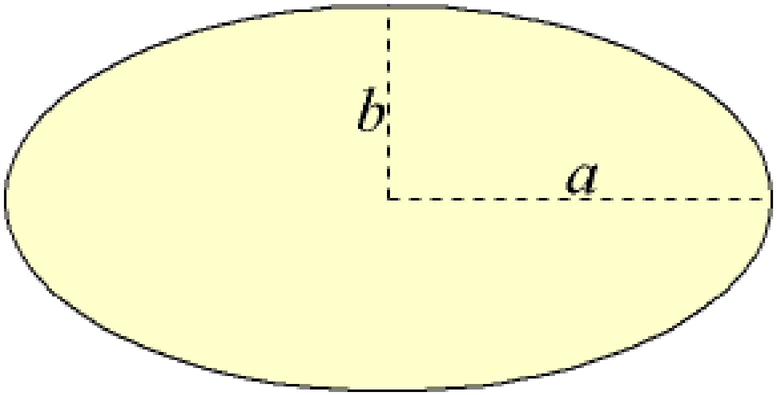
- Integrate function over a domain

$$\iint_{(x,y)\in A} dx\,dy\,f(x,y)$$



Simple example: Area of Ellipse

- Classical formula: $A = \pi a b$



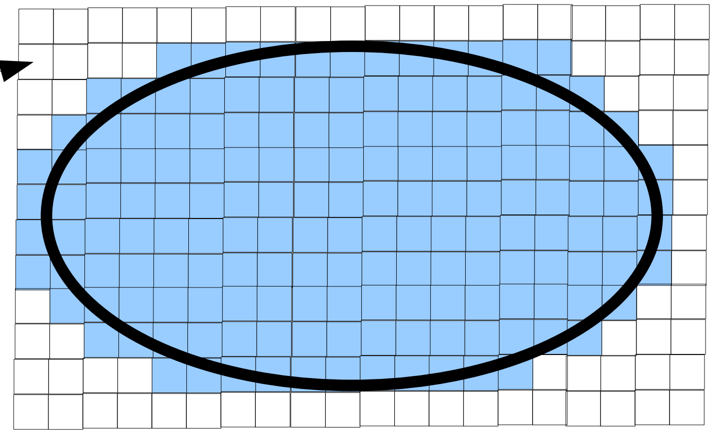
- Points inside ellipse satisfy:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} < 1$$

- Finding area is same as integrating over domain of function f(x, y) = 1.

Area of ellipse: count boxes

Area of little box
= dx dy



- Algorithm:

1. Initialize: A = 0
2. Loop over all boxes
3. For each box, if $\frac{x^2}{a^2}+\frac{y^2}{b^2}<1$ then
   A = A + dx dy
4. End loop
5. Return A

The core of this algorithm involves knowing when we're inside or outside the boundary

If box is inside, add in area of small box.

Matlab code

```
function A = box_count_ellipse(a, b)
  % This fcn uses box counting to get the area of an
  % ellipse whose semi-major/minor axes are a, b.

  N = 10000;
  x = linspace(-a, a, N);
  y = linspace(-b, b, N);
  dx = x(2) - x(1);
  dy = y(2) - y(1);


  % Convenience definitions.
  a2 = a*a;
  b2 = b*b;

  A = 0;          % Initialize A
  for ix = 1:N
    for iy = 1:N
      p = x(ix)*x(ix)/a2 + y(iy)*y(iy)/b2;
      if (p < 1)
        A = A + dx*dy;
      end
    end
  end

end
```

Include f(x, y) here if integrating a more complicated function than f(x, y) = 1.

Box counting results

- N = 100

```
>> test_ellipse_area_boxcount
Ellipse a, b = 1.000000, 1.000000 ...
Acomputed = 3.129477, Atrue = 3.141593, reldiff = -3.856665e-03

Ellipse a, b = 3.672186, 1.130595 ...
Acomputed = 12.992818, Atrue = 13.043121, reldiff = -3.856665e-03
```

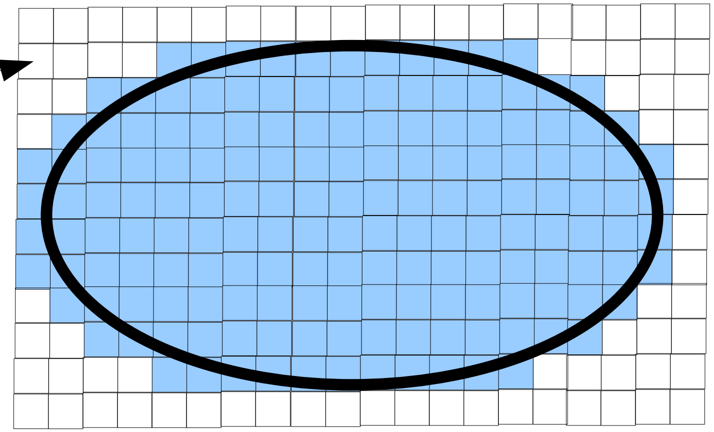
- N = 10000

```
>> test_ellipse_area_boxcount
Ellipse a, b = 1.000000, 1.000000 ...
Acomputed = 3.141583, Atrue = 3.141593, reldiff = -3.172019e-06

Ellipse a, b = 1.952509, 0.541740 ...
Acomputed = 3.323019, Atrue = 3.323029, reldiff = -3.172641e-06
```

Box counting for ellipse....

Area of little box = dx dy



1. Initialize: A = 0
2. Loop over all boxes
3. For each box, if $\frac{x^2}{a^2}+\frac{y^2}{b^2}<1$ then

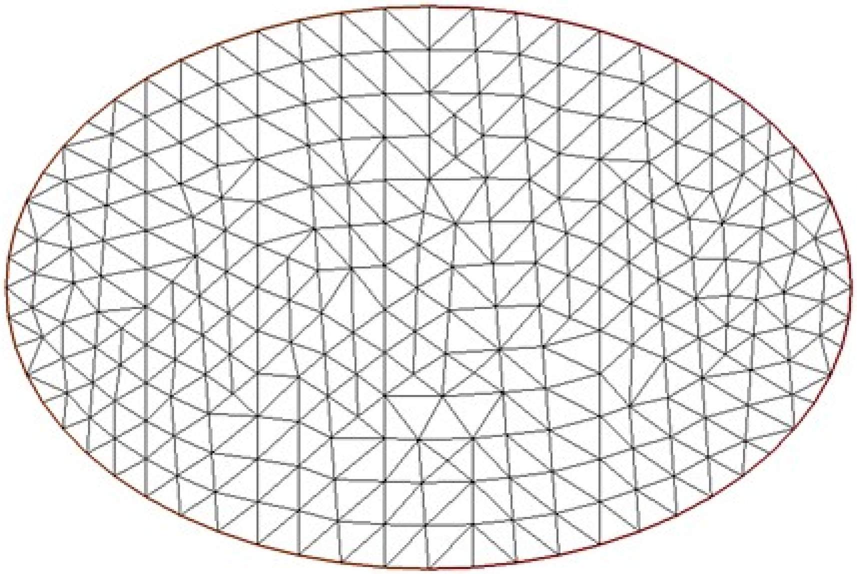
   A = A + dx dy
4. End loop
5. Return A

The core of this algorithm involves knowing when we're inside or outside the boundary

But accuracy stinks because boundaries are not well handled

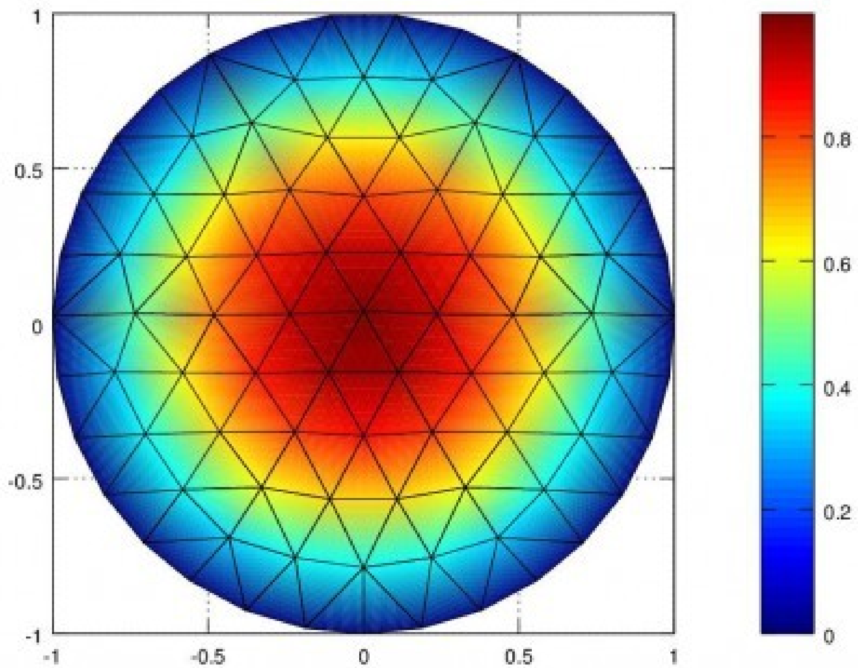
A different method

- Mesh your object, then compute area of each triangle.



- More accurate than counting boxes – computation is 2D trapezoidal method.
- Boundaries are handled better.
- Method fits nicely with data structures and concepts used in FEA analysis.

Concept: mesh



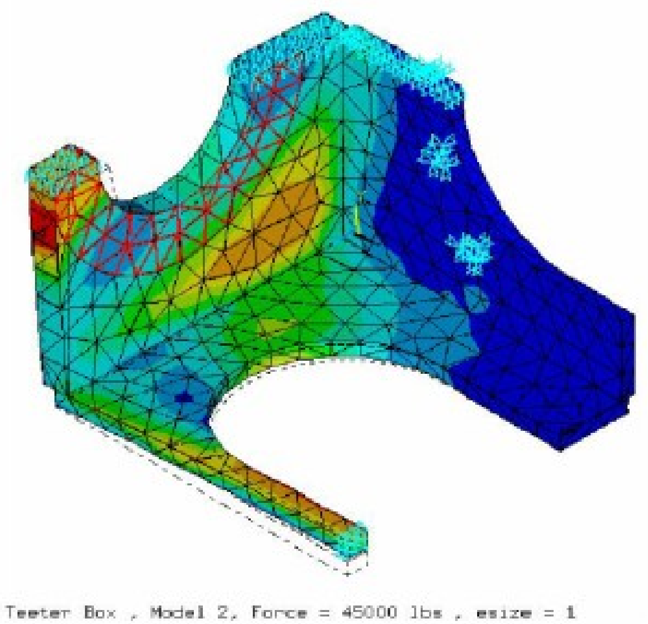
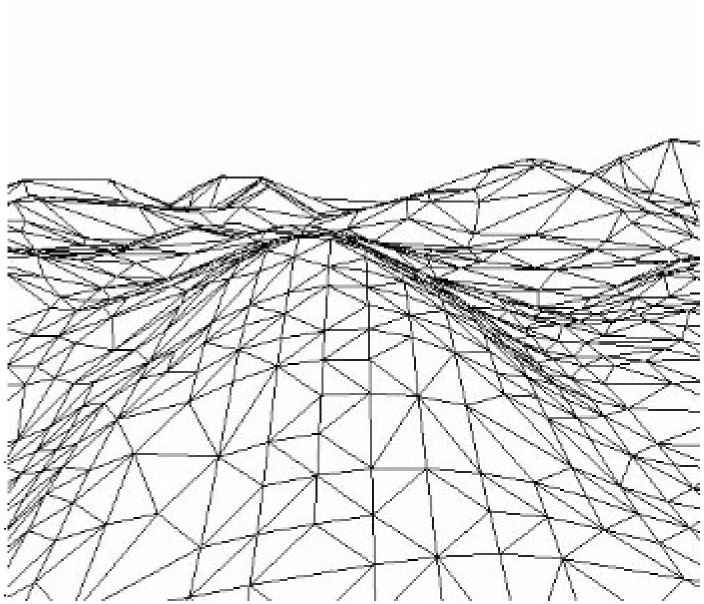


Figure 3
http://www.umass.edu/mie/labs/mda/fea/fealib/goldstein/PROJECT.html

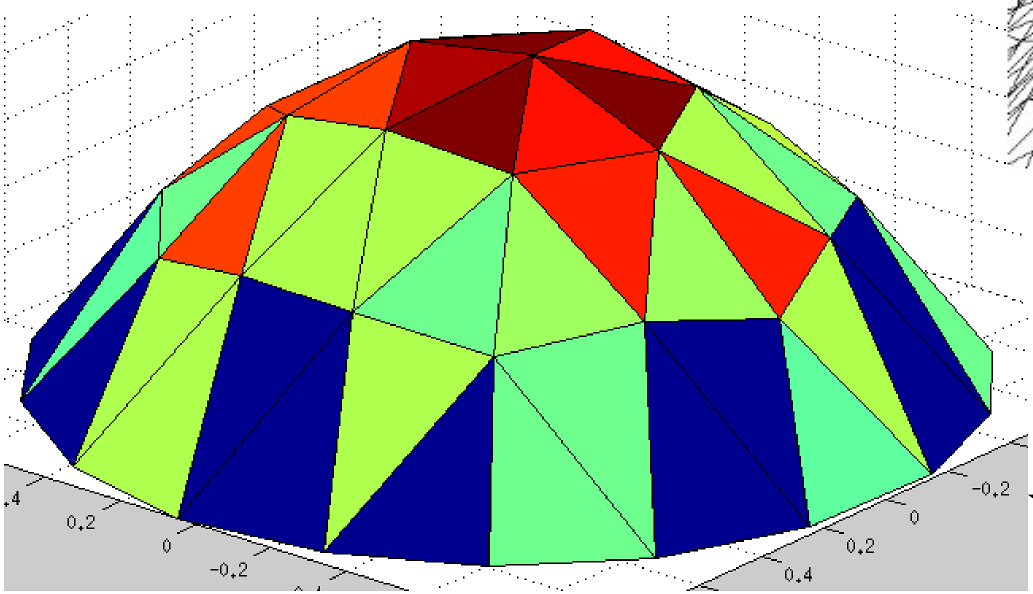
- Cover domain with triangles.
- Triangles follow boundaries much better than simple squares.

Representing a function using a mesh

- Assemble individual triangles into function defined over domain



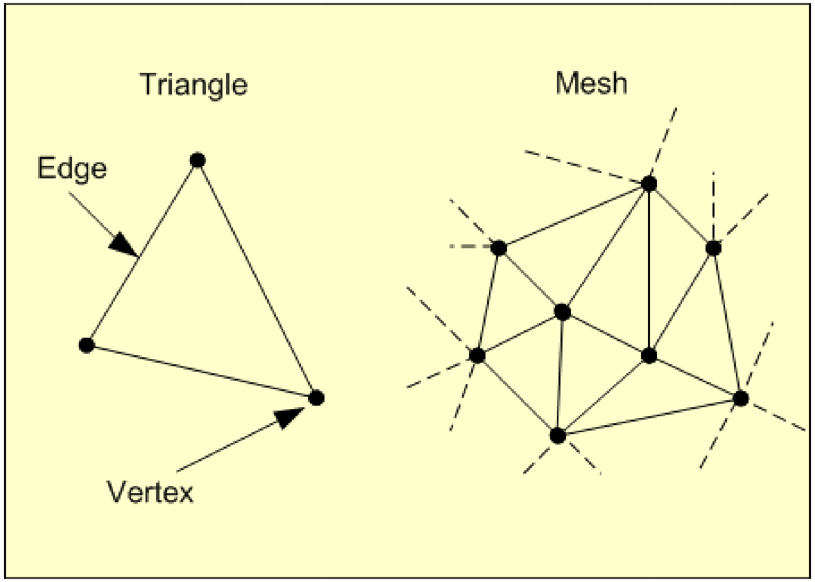
Arbitrary function



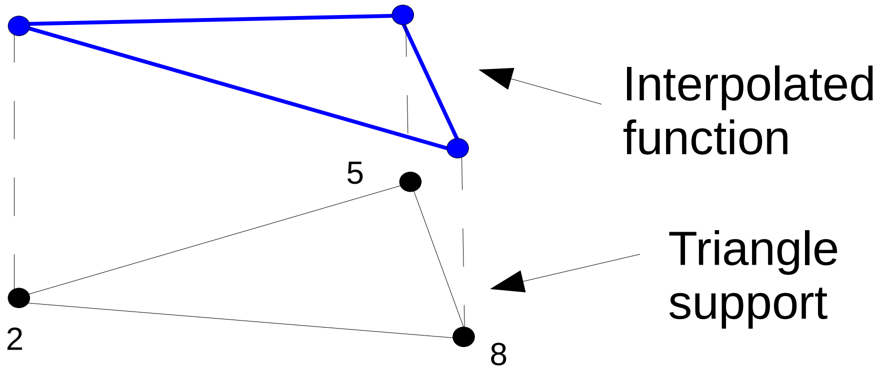
$$1-\left(x^2+y^2\right)$$

Using a mesh to represent a 2D function

- Cover domain with triangles.
- Triangles have vertices (points).
- Triangles have edges (line segments).

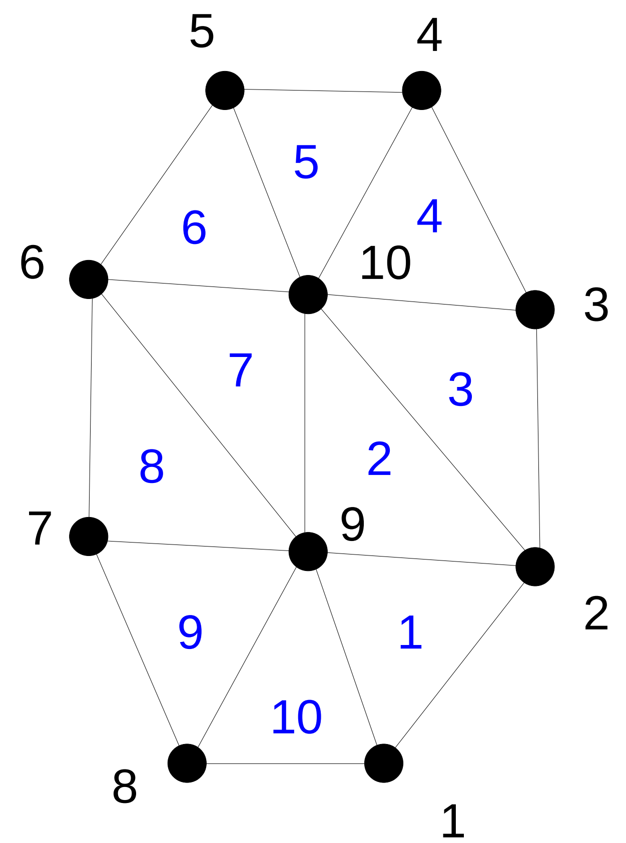


- Use linear interpolation to find value of function inside triangle



Representing a mesh in Matlab

eIdx	Elements (triangles)			nIdx	Nodes (vertices)	
1	1	2	9	1	x1	y1
2	9	2	10	2	x2	y2
3	2	3	10	3	x3	y3
4	10	3	4	4	x4	y4
5	4	5	10	5	x5	y5
6	5	6	10	6	x6	y6
7	6	9	10	7	x7	y7
8	6	7	9	8	x8	y8
9	7	8	9	9	x9	y9
10	8	1	9	10	x10	y10

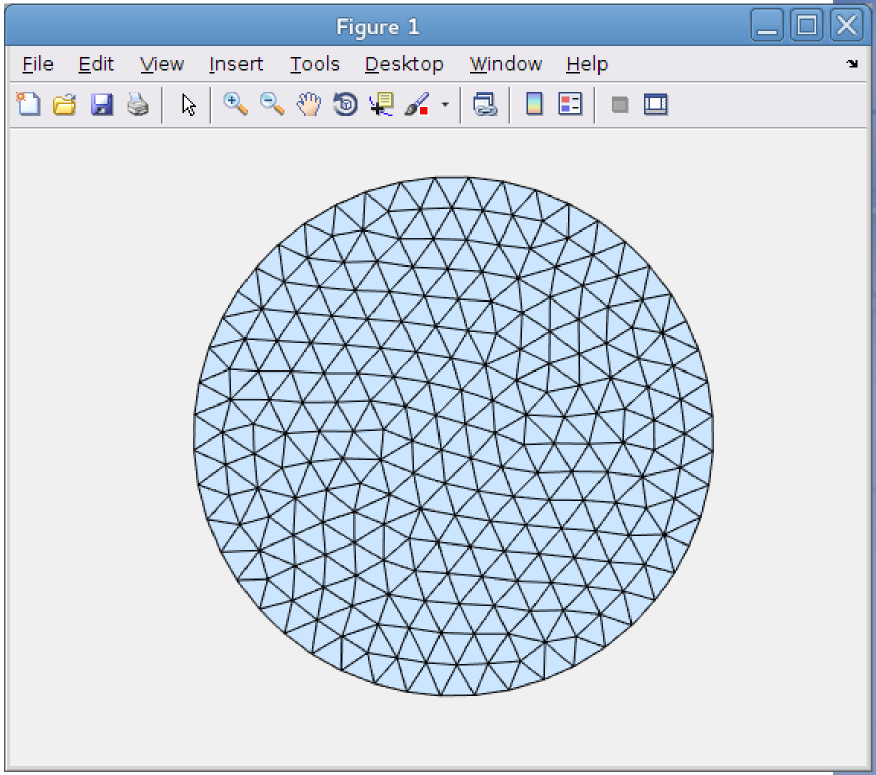


Nodes are black
Triangles (elements) are blue

Note all nodes listed
in CCW order.

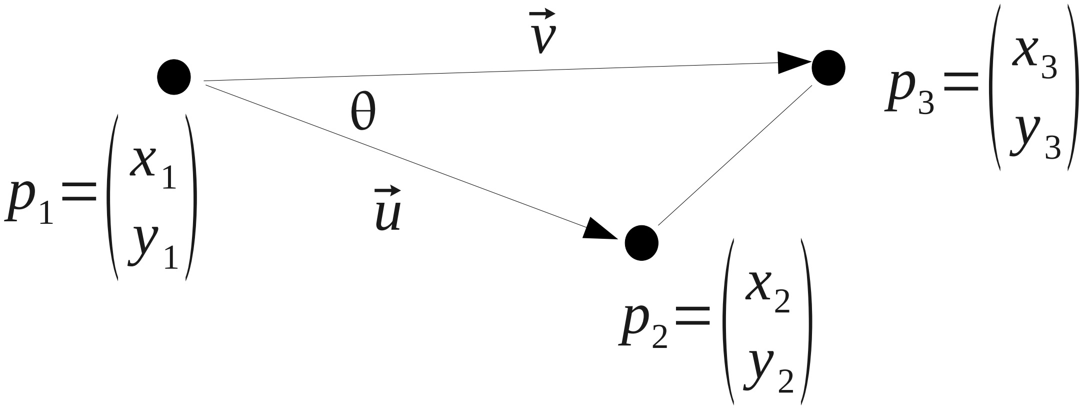
Example: Find area of circle in 2D

- Inputs:
  - Meshed circle (P and T matrices)
- Desired output:
  - Sum of areas of all triangles
- Question:
  - How to compute area of each triangle?



Area of triangle

- Triangle is defined by three vertex points.



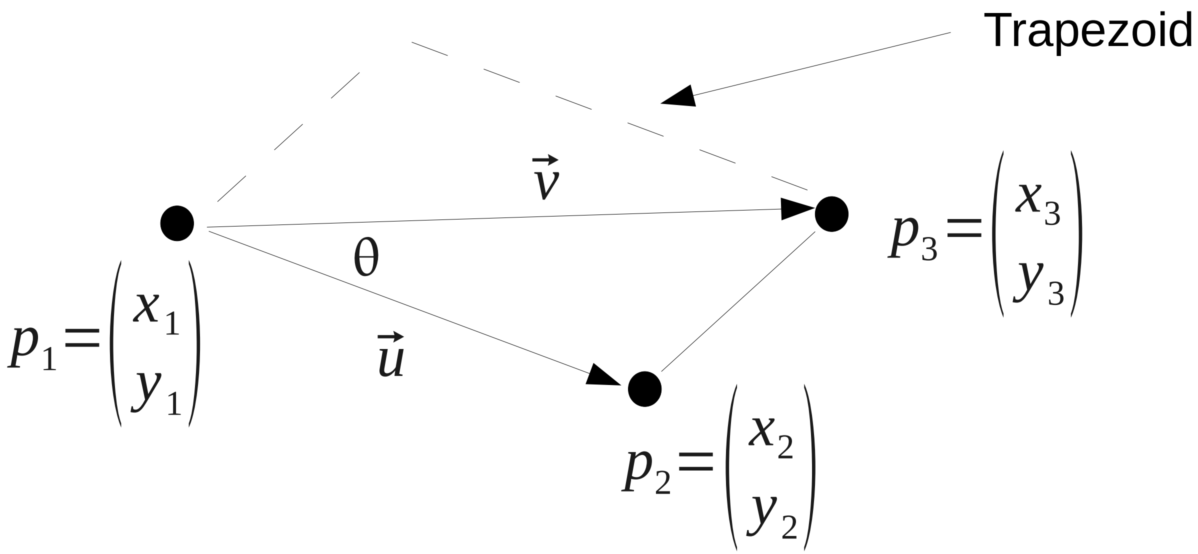
- Consider vectors u, v, defined by

$$\vec{u} = \vec{p}_2 - \vec{p}_1 \qquad \vec{v} = \vec{p}_3 - \vec{p}_1$$

- Recall formula for vector cross product:

$$\vec{u} \times \vec{v} = |u||v| \sin(\theta)\, \hat{z}$$

Triangle area



- Area of trapezoid

$$A_{trap} = \vec{u} \times \vec{v} = |u||v| \sin(\theta)$$

- Area of triangle:

$$A_{Tri} = \frac{1}{2} |u||v| \sin(\theta)$$

Cross product

- Recall another definition:

$$\vec{u}\times\vec{v}=\begin{vmatrix}\hat{x} & \hat{y} & \hat{z}\\ u_x & u_y & u_z\\ v_x & v_y & v_z\end{vmatrix}$$

Evaluate this like a determinant

- From last slide we have

$$\vec{u}=\begin{pmatrix}x_2-x_1\\ y_2-y_1\end{pmatrix} \qquad \vec{v}=\begin{pmatrix}x_3-x_1\\ y_3-y_1\end{pmatrix}$$

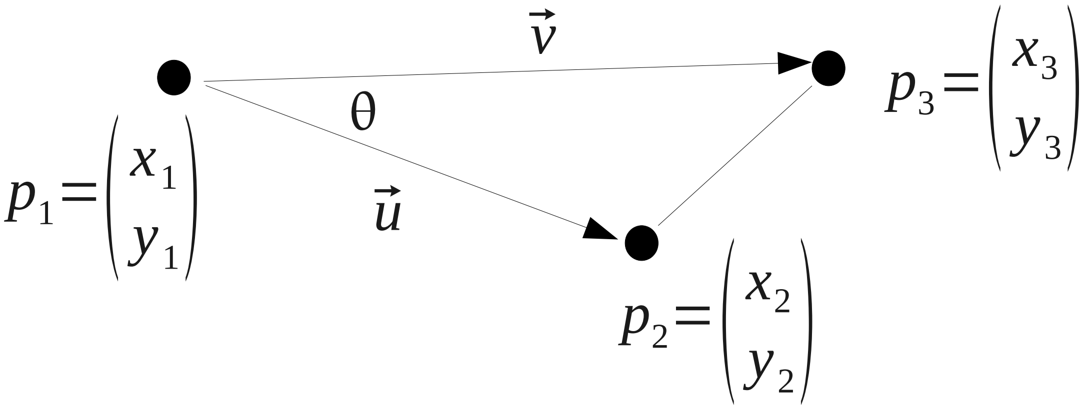
- Insert into det:

$$\vec{u}\times\vec{v}=\begin{vmatrix}\hat{x} & \hat{y} & \hat{z}\\ x_2-x_1 & y_2-y_1 & 0\\ x_3-x_1 & y_3-y_1 & 0\end{vmatrix}$$

- We care about $\hat{z}$ component:

$$=(x_2-x_1)(y_3-y_1)-(y_2-y_1)(x_3-x_1)$$

Area of triangle



$$A_{Tri} = \frac{1}{2}\left((x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)\right)$$

$$= \frac{1}{2}\begin{vmatrix} (x_2 - x_1) & (x_3 - x_1) \\ (y_2 - y_1) & (y_3 - y_1) \end{vmatrix}$$

"Surveyor's area formula"

Area-finding algorithm

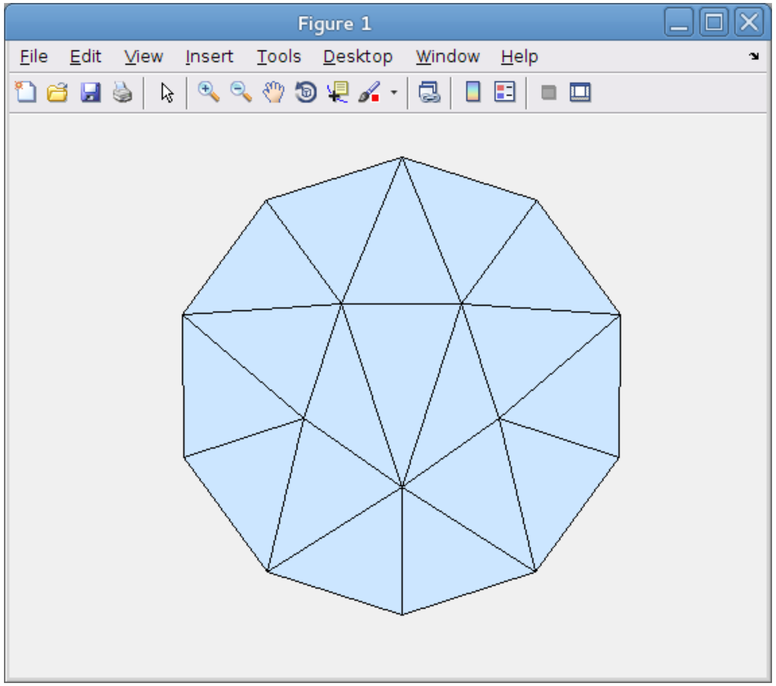
1. Initialize Atot = 0.
2. Loop over all triangles.
3. For each triangle t, use surveyer's area formula to compute area Atri
4. Atot = Atot + Atri
5. Continue looping -- go back to 2
6. At end of looping, return Atot.

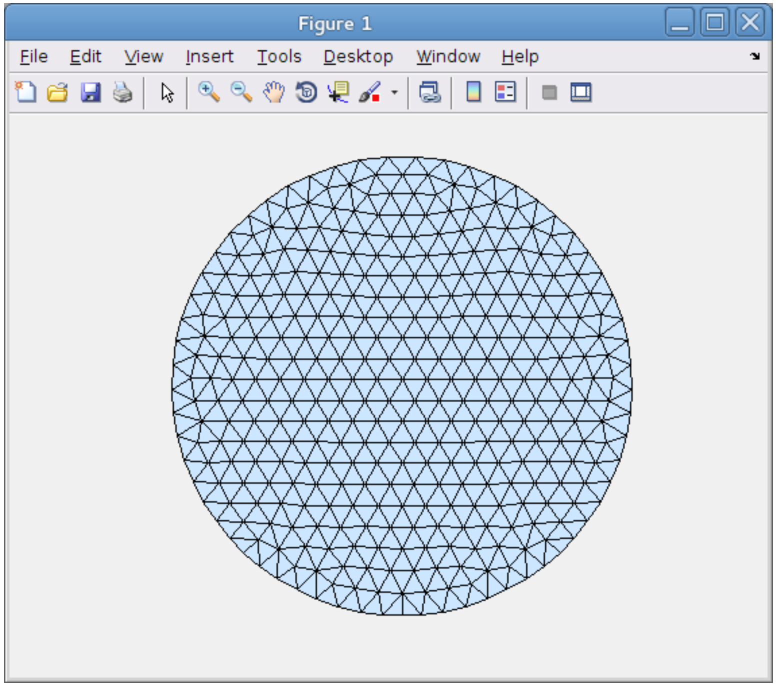
```
% Perform integration by summing over triangles
s = 0;
for idx = 1:size(T, 1)
  p1 = P(T(idx, 1), :)';
  p2 = P(T(idx, 2), :)';
  p3 = P(T(idx, 3), :)';
  s = s + tri_area_surveyers(p1, p2, p3);
end
```

```
function A = tri_area_surveyers(p1, p2, p3)
  % Inputs:
  % p1, p2, p3 = vertices of triangle as 2D vectors [x, y]'
  %
  % Outputs:
  % A = computed area of triangle [p1, p2, p3]
  %
  % Area is computed using the Surveyer's triangle
  % area formula A = (1/2)*det([ux vx; uy vy]) where
  % u = p2 - p1, and v = p3 - p1.

  u = p2 - p1;
  v = p3 - p1;
  S = [u(1) v(1); u(2) v(2)];
  A = abs(det(S))/2;
end
```

Area of unit circle

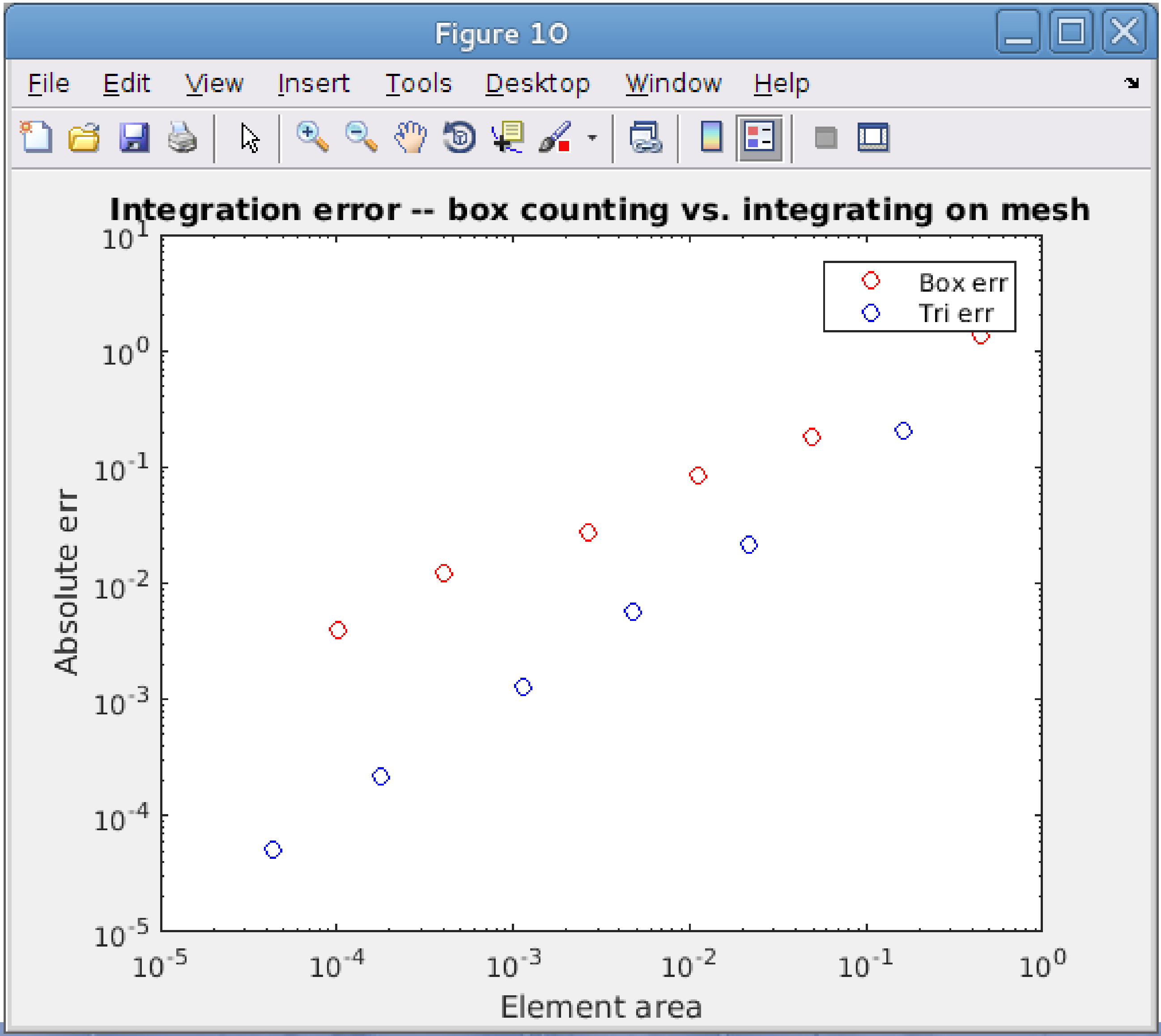





```
tri size param = 0.500000,
Atrue = 3.141593,
Asurveyers = 2.938924,
abs error = 2.026682e-01
```

```
tri size param = 0.100000,
Atrue = 3.141593,
Asurveyers = 3.135911,
abs error = 5.681554e-03
```

Compare box counting to integration on triangular mesh

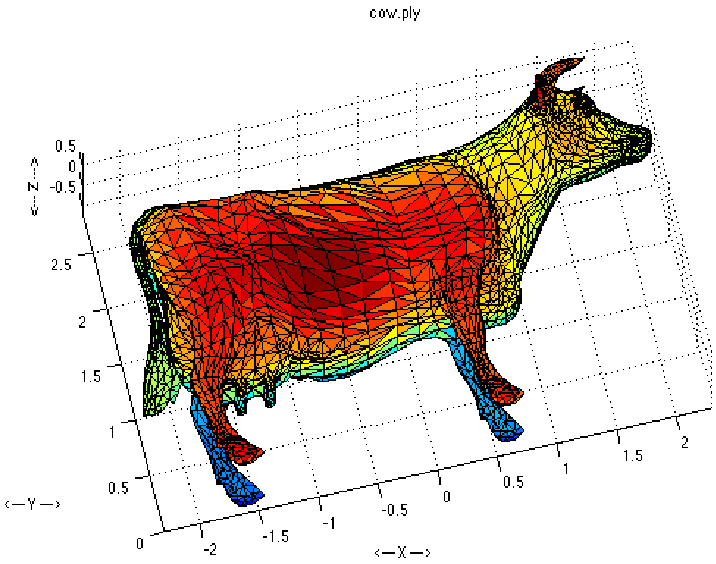


$$err(box) \sim \sqrt{A}$$

$$err(tri) \sim A$$

More complicated shapes?

- Usually, the mesh is given to you.
- Use a program to create a suitable mesh for your problem.
- Many file formats:
  - STL
  - PLY
  - Etc...



- If mesh is regular enough, you can make it yourself.

Next: integrating the volume under a surface

- Find surface area of a domain

$$\iint_{(x,y)\in A} dx\,dy\,1$$

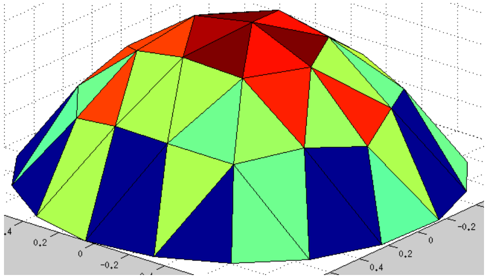
We just did this

- Integrate function over a domain

$$\iint_{(x,y)\in A} dx\,dy\,f(x,y)$$

What about doing this?

Integrate a function over a meshed domain

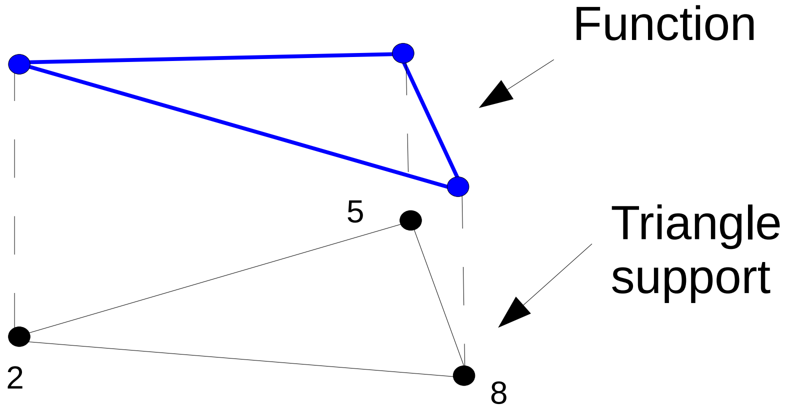


- Global integral is found as sum of integrals over triangles

$$\iint_{(x,y)\in A} dx\,dy\, f(x,y) = \sum_i \iint_{(x,y)\in A_i} dx\,dy\, f(x,y)$$

- Question: How to compute integral over each triangle?

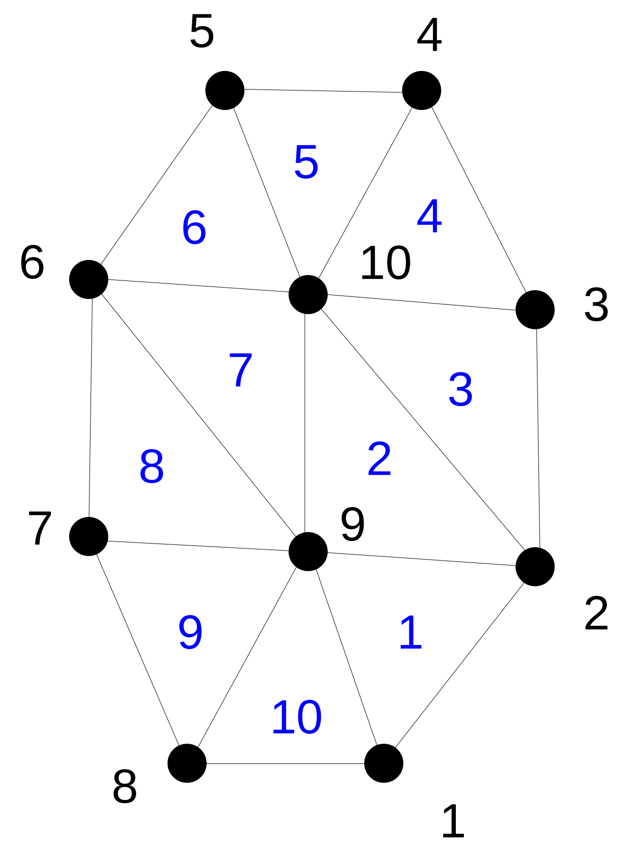
$$\iint_{(x,y)\in A_i} dx\,dy\, f(x,y)$$



Representing a mesh in Matlab

eIdx	Elements (triangles)		
1	1	2	9
2	9	2	10
3	2	3	10
4	10	3	4
5	4	5	10
6	5	6	10
7	6	9	10
8	6	7	9
9	7	8	9
10	8	1	9

nIdx	Nodes (vertices)	
1	x1	y1
2	x2	y2
3	x3	y3
4	x4	y4
5	x5	y5
6	x6	y6
7	x7	y7
8	x8	y8
9	x9	y9
10	x10	y10

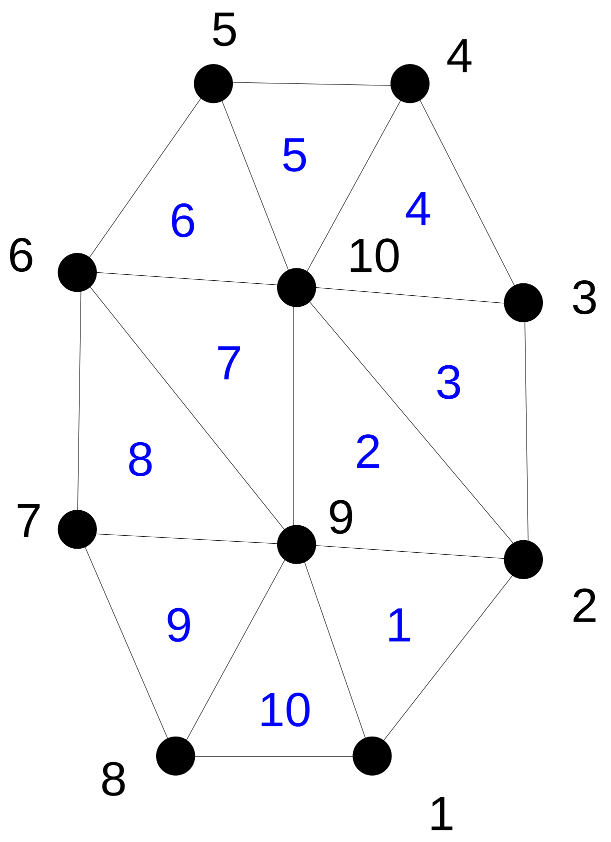


Nodes are black
Triangles (elements) are blue

Note all nodes listed
in CCW order.

New: Function vector

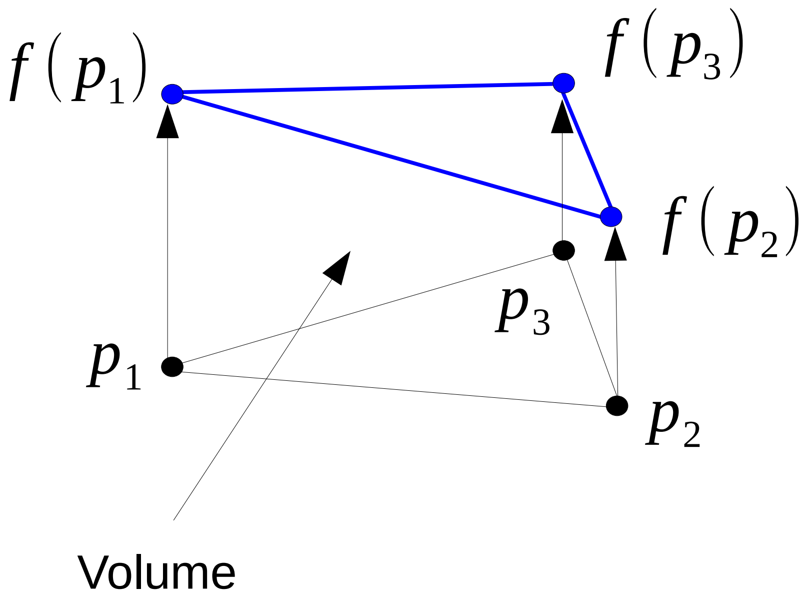
nIdx	f(x, y)
1	U1
2	U2
3	U3
4	U4
5	U5
6	U6
7	U7
8	U8
9	U8
10	U10



Function vector f(x, y) is defined on the mesh nodes (vertices).

Inputs and outputs

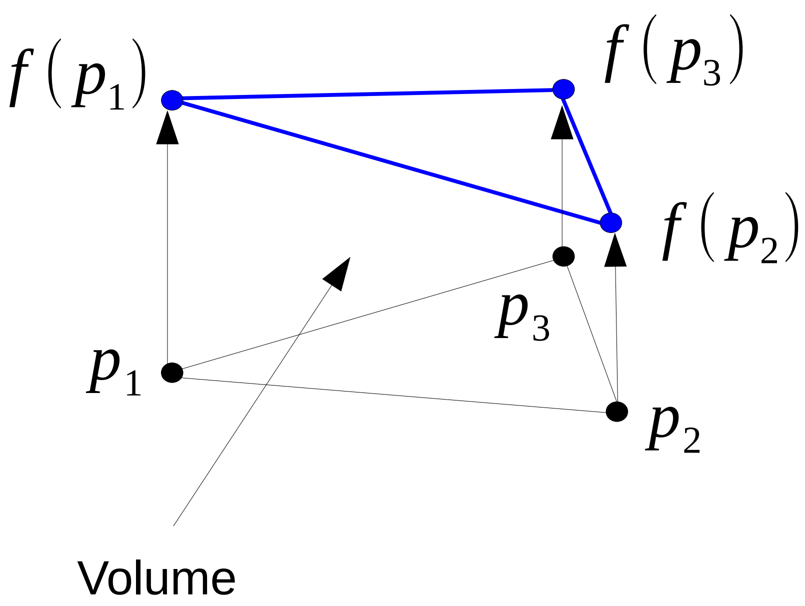
- Inputs at each triangle:
  - Locations of vertices p_1, p_2, p_3.
  - Function value at vertices, $f(p_1)$, $f(p_1)$, $f(p_1)$,
- Assumption:
  - Blue triangle is planar.
- Desired output:
  - Volume underneath top triangle



Big problem

- If vertices are at arbitrary positions in plane, how to set limits of integration?

$$V = \int_{x_0}^{x_1} \int_{y_0}^{y_1} f(x, y)\, dy\, dx$$

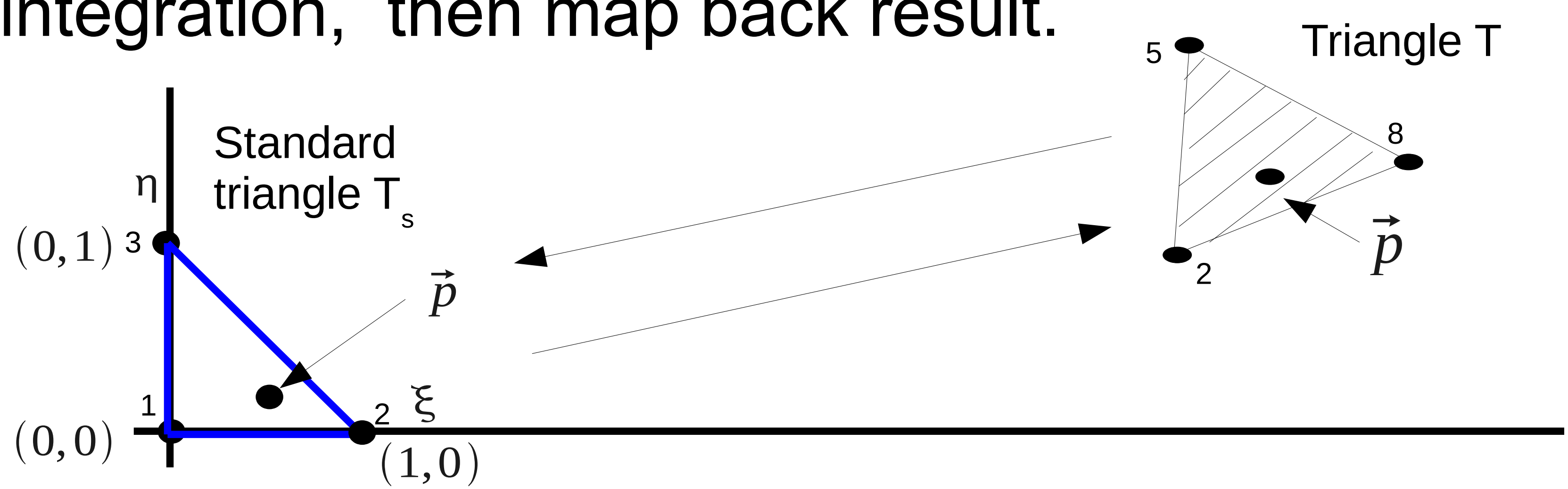


- It’s not obvious …

Concept: standard triangle

- How to perform integral of function over some arbitrary triangular patch?

- Map triangle back to “standard triangle”, do integration, then map back result.



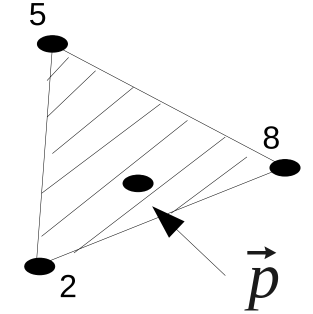
Mapping between triangles

- Note any point in T may be written using barycentric coordinates:

$$\vec{p} = (1-\xi-\eta)\vec{p}_2 + \xi\vec{p}_8 + \eta\vec{p}_5$$

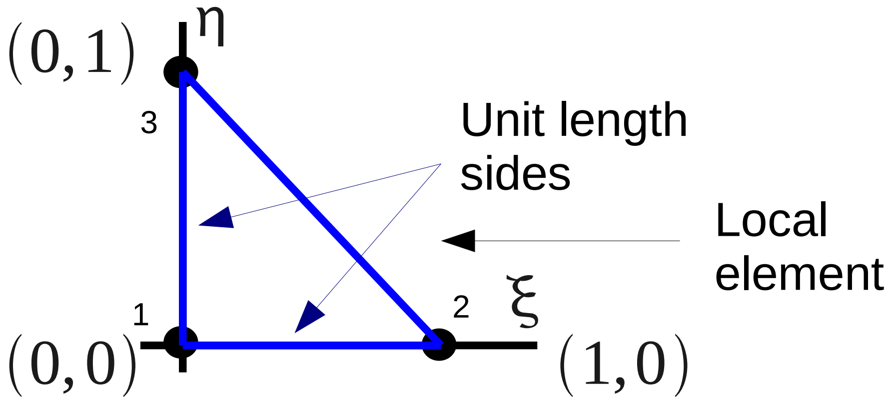
With $\eta \in [0,1]$ $\xi \in [0, 1-\eta]$

Eta Xi



Point in global mesh

- This induces a mapping to the “standard” triangle T_s:



Local vertex $\Leftrightarrow$ Global vertex

$1 \Leftrightarrow 2$

$2 \Leftrightarrow 8$

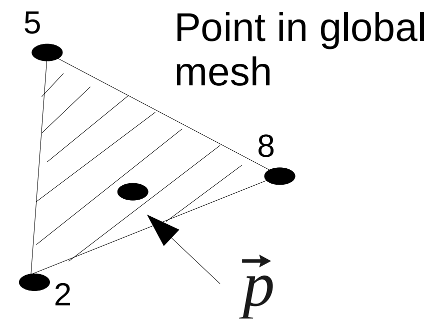
$3 \Leftrightarrow 5$

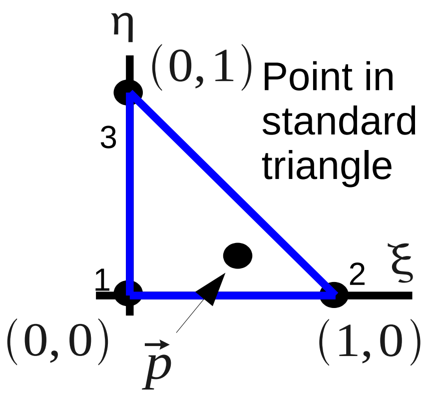
Mapping between triangles

- Jacobian matrix associated with coordinate change encodes scale change

$$\vec{p} = (1-\xi-\eta)\,\vec{p}_2 + \xi\,\vec{p}_8 + \eta\,\vec{p}_5$$

$$= \vec{p}_1 + \xi(\vec{p}_2 - \vec{p}_1) + \eta(\vec{p}_3 - \vec{p}_1)$$





$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} p_{1x} \\ p_{1y} \end{pmatrix} + \xi \begin{pmatrix} p_{2x} - p_{1x} \\ p_{2y} - p_{1y} \end{pmatrix} + \eta \begin{pmatrix} p_{3x} - p_{1x} \\ p_{3y} - p_{1y} \end{pmatrix}$$

- Jacobian

$$J = \left| \frac{\partial(x, y)}{\partial(\xi, \eta)} \right| = \det \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{vmatrix}$$

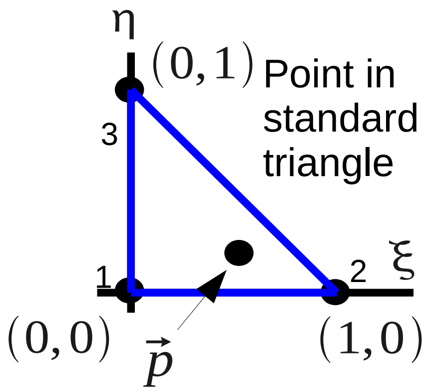
Handles scale change due to coordinate change

Evaluate integral

- Compute integral on standard triangle:

$$\iint_{(x,y)\in A_i} dx\,dy\,f(x,y)$$

$$=\int_0^1 d\eta \int_0^{(1-\eta)} d\xi \left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right| f(\eta,\xi)$$



- Determinant is a constant scale factor:

$$J=(p_{2x}-p_{1x})(p_{3y}-p_{1y})-(p_{3x}-p_{1x})(p_{2y}-p_{1y})$$

- Look familiar? This is 2x area of original triangle. That is,

$$J=\left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right|=2A_T$$

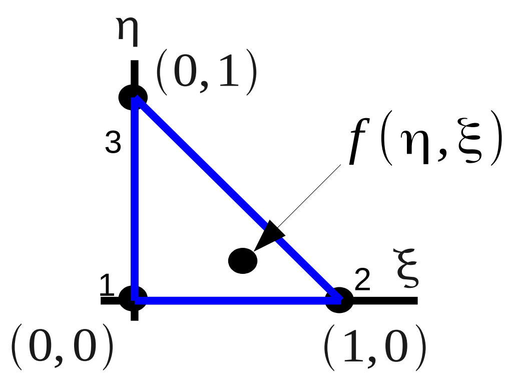
Computing area under triangle

- From knowing the Jacobian, we have

$$\iint_{(x,y)\in A_i} dx\,dy\,f(x,y)$$

$$=\int_0^1 d\eta \int_0^{(1-\eta)} d\xi \left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right| f(\eta,\xi)$$

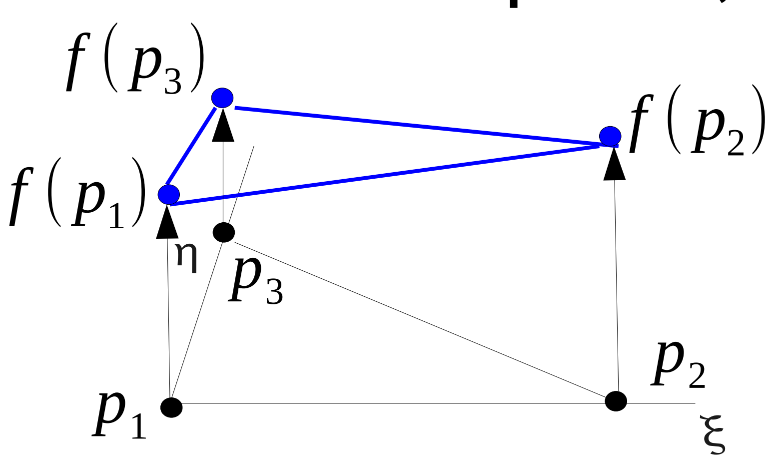
$$=2A_i\int_0^1 d\eta \int_0^{(1-\eta)} d\xi\, f(\eta,\xi)$$



- What is $f(\eta,\xi)$? Since function is a plane,

$$f(\eta,\xi)=a\xi+b\eta+c$$

Recall interpolation on a triangle from class 9

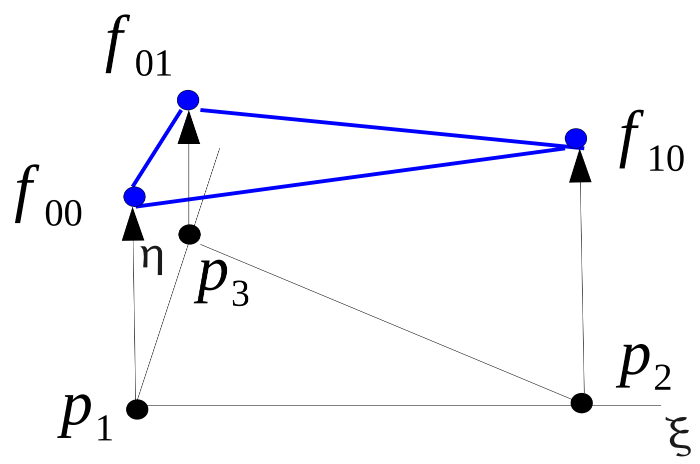


- Function is a plane

$$f(\eta,\xi)=a\xi+b\eta+c$$

- We know the function values at the vertices:

$$f_{00}=a\cdot 0+b\cdot 0+c$$
$$f_{10}=a\cdot 1+b\cdot 0+c$$
$$f_{01}=a\cdot 0+b\cdot 1+c$$



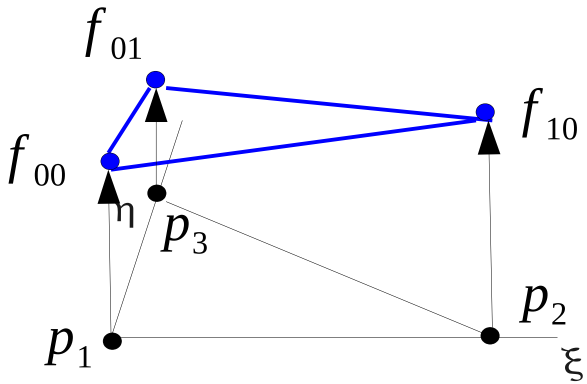
- Written in matrix form:

$$\begin{pmatrix}0 & 0 & 1\\ 1 & 0 & 1\\ 0 & 1 & 1\end{pmatrix}\begin{pmatrix}a\\ b\\ c\end{pmatrix}=\begin{pmatrix}f_{00}\\ f_{10}\\ f_{01}\end{pmatrix}$$

Since we know the matrix, and we know f, we can get [a, b, c] using a matrix solve operation.

Next, perform integration on standard triangle

$$\iint_{(x,y)\in A_i} dx\,dy\,f(x,y)$$



$$=2A_i\int_0^1 d\eta \int_0^{(1-\eta)} d\xi\left(a\xi+b\eta+c\right)$$

$$=2A_i\int_0^1 d\eta\left(\frac{a}{2}\xi^2+b\eta\xi+c\xi\right)\Bigg|_0^{1-\eta}$$

$$=2A_i\int_0^1 d\eta\left(\frac{a}{2}(1-\eta)^2+b\eta(1-\eta)+c(1-\eta)\right)$$

Split into three pieces

$$2A_i\int_0^1 d\eta\left(\frac{a}{2}(1-\eta)^2+b\eta(1-\eta)+c(1-\eta)\right)$$

$$2A_i\int_0^1 d\eta\left(\frac{a}{2}(1-\eta)^2\right)$$

Substitute:

$$s=1-\eta \quad ds=-d\eta$$

Integral becomes:

$$-2A_i\int_1^0 ds\left(\frac{a}{2}s^2\right)$$

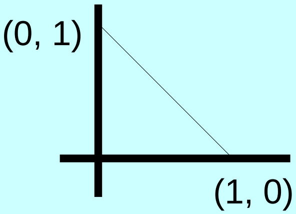
$$=2A_i\left(\frac{a}{6}\right)$$

$$2A_i\int_0^1 d\eta\left(b\eta(1-\eta)\right)$$

$$=2A_i\left(\frac{1}{2}\eta^2-\frac{1}{3}\eta^3\right)\Big|_0^1$$

$$=2A_ib\left(\frac{1}{2}\eta^2-\frac{1}{3}\eta^3\right)\Big|_0^1$$

$$=2A_i\left(\frac{b}{6}\right)$$

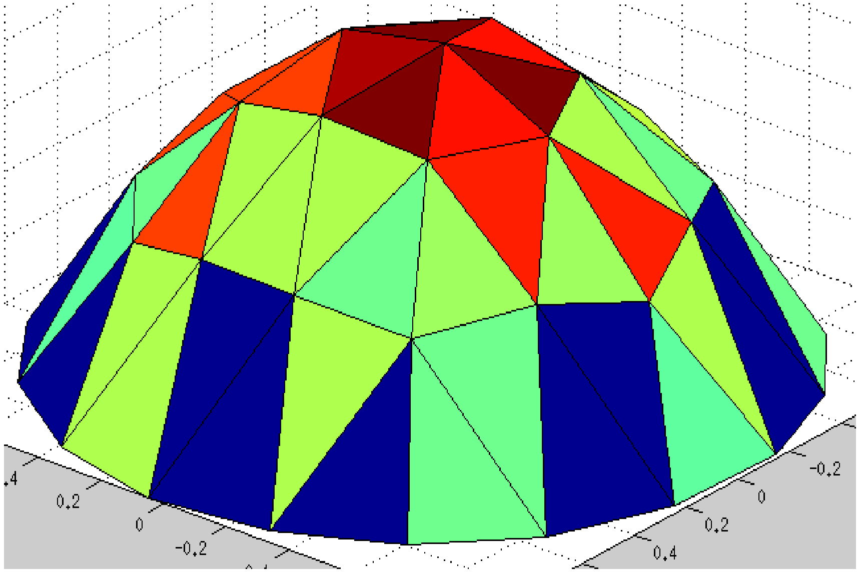


$$=2A_i\left(\frac{c}{2}\right)$$

Integrating over meshed domain: Putting it all together...

1. Loop over all triangles.

2. For each triangle, send $p_1, p_2, p_3, f_1, f_2, f_3$ to sub-fcn.



3. Compute Jacobian

$$J = 2A_T = (p_{2x} - p_{1x})(p_{3y} - p_{1y}) - (p_{3x} - p_{1x})(p_{2y} - p_{1y})$$

4. Compute [a, b, c] using linear solve for a, b, c coeffs:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \backslash \begin{pmatrix} f_{00} \\ f_{10} \\ f_{01} \end{pmatrix}$$

$$f(\eta, \xi) = a\xi + b\eta + c$$

Recall interpolation on a triangle from class 9

5. Compute integral on standard triangle:

$$\text{Integral} = \left(\frac{a}{6}\right) + \left(\frac{b}{6}\right) + \left(\frac{c}{2}\right)$$

6. Multiply by Jacobian, then return.

Result is integral over one triangle in mesh.

$$\iint_{(x,y)\in A_i} dx\,dy\,f(x,y) = J\left[\left(\frac{a}{6}\right) + \left(\frac{b}{6}\right) + \left(\frac{c}{2}\right)\right]$$

7. Add returned integral over single triangle to global sum.

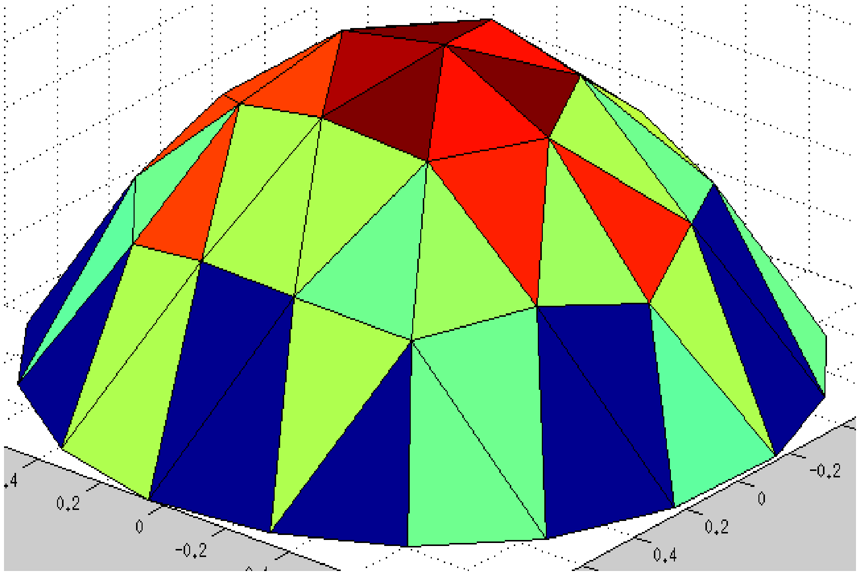
$$\iint_{(x,y)\in A} dx\,dy\,f(x,y) = \sum_i \iint_{(x,y)\in A_i} dx\,dy\,f(x,y)$$

8. When done looping, the global sum is the integral of the function over the triangulated domain.

Summary: Integrating meshed function over irregular domain

$$\iint_{(x,y)\in A} dx\,dy\,f(x,y) = \sum_i \iint_{(x,y)\in A_i} dx\,dy\,f(x,y)$$

- Multi-step process
- Sum volumes of triangular domains.
- Replace function over patch with interpolant.



Comments

- The classical methods involving sampling at a few points (Gauss quadrature) reflect their time period.
  - Hand computation was very expensive.

- The computer allows us to use meshing/triangulation as an better way to integrate over irregular domains.

Session summary

- Integration in 2 dimensions
  - Box counting.
  - 2D Gaussian quadrature.
- Integrating shapes over odd boundaries
  - Box counting.
  - Meshing using triangles.
  - Integrating over a triangle mesh.