ასტროფიზიკის და პლაზმის ფიზიკის ამოცანების მოდელირება - 2

ლექცია 7

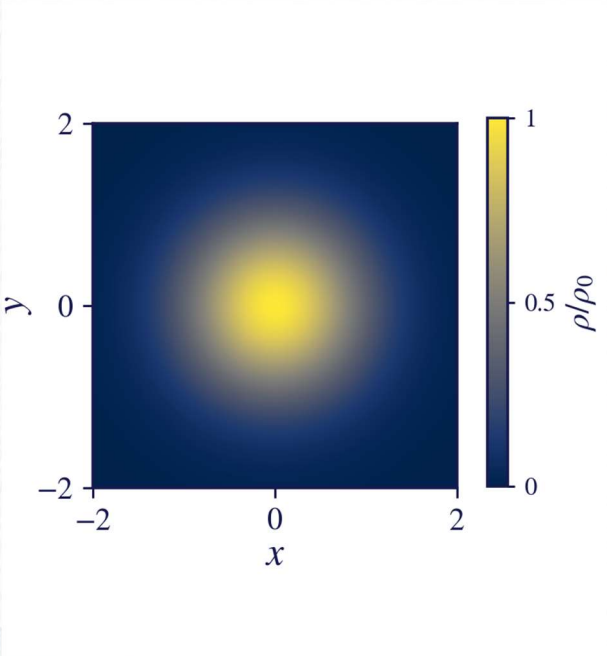
Scientific Visualization

Techniques for Modelling Physical Fields

Scalar, Vector and Tensor Fields

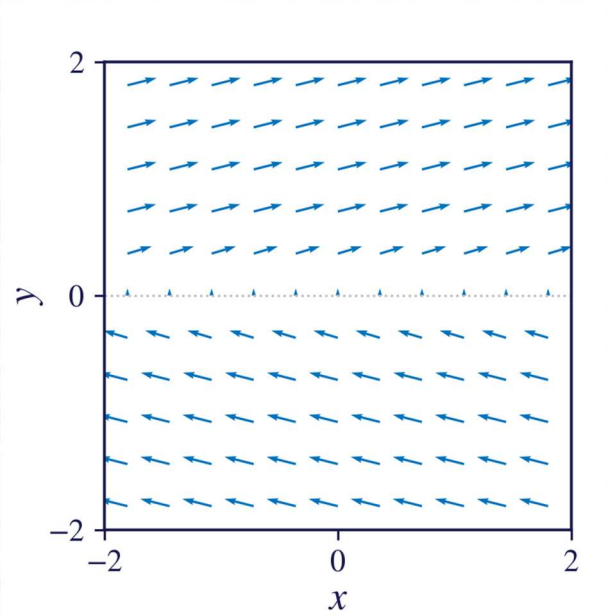
Choose the representation from the quantity stored at each point.

Scalar field



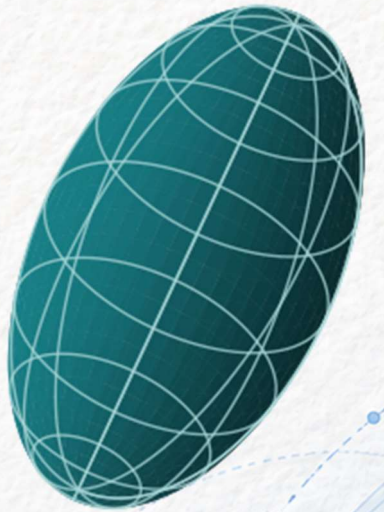
ρ(x, y, z)

Vector field



B(x, y, z)

Tensor field



P(x, y, z)

Field type is independent of spatial dimension and time dependence.

From Model Array to Figure

Keep data preparation separate from graphical representation.

1. Data	2. Representation	3. Display
Coordinates + values X, Y, Rho	→ Map, contour, surface imagesc / contour / surf	→ Axes, scale, camera axis / clim / view

MATLAB

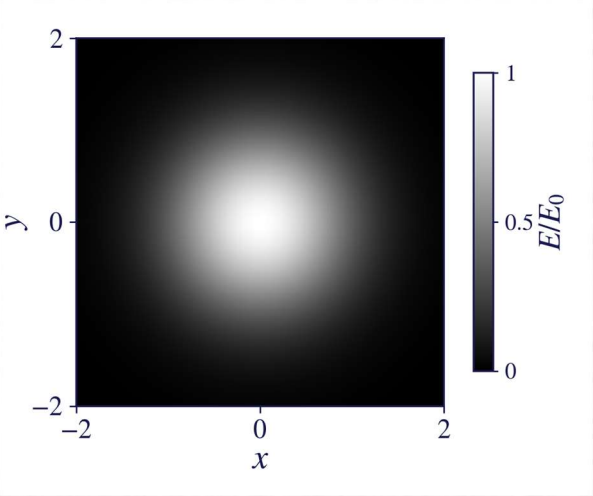
```
x = linspace(-2,2,101);
[X,Y] = meshgrid(x,x);
Rho = exp(-(X.^2 + Y.^2));
imagesc(x,x,Rho); axis xy image; colorbar
```

For meshgrid: rows follow y, columns follow x; name every plotted quantity.

Consistent Display Scales

Compare a reference field with a ten-times weaker field.

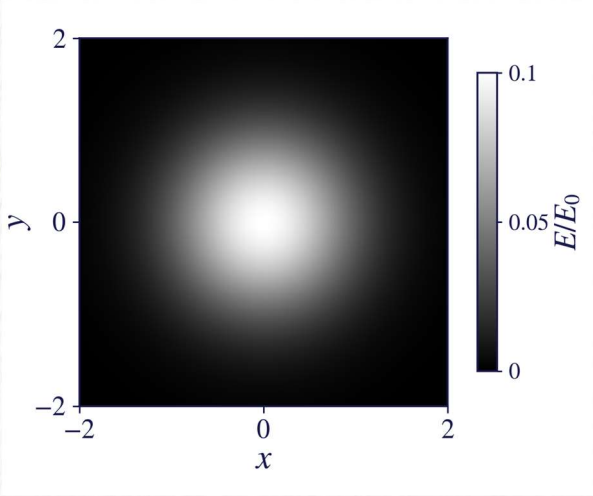
Reference



MATLAB

```
imagesc(x,y,Ei)
clim([0 1])
```

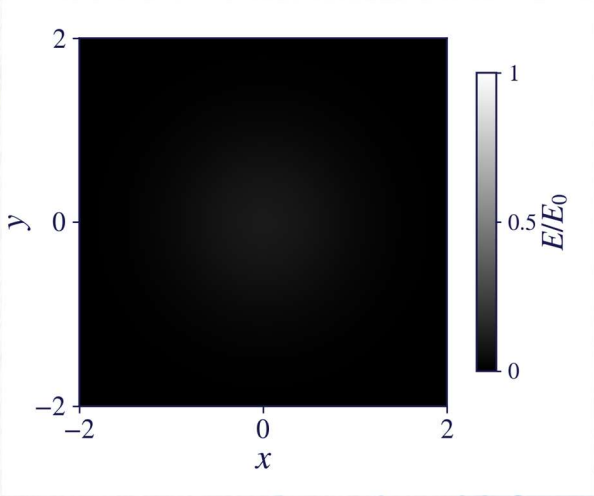
Automatic limits



MATLAB

```
imagesc(x,y,0.1*Ei)
clim('auto')
```

Fixed limits



MATLAB

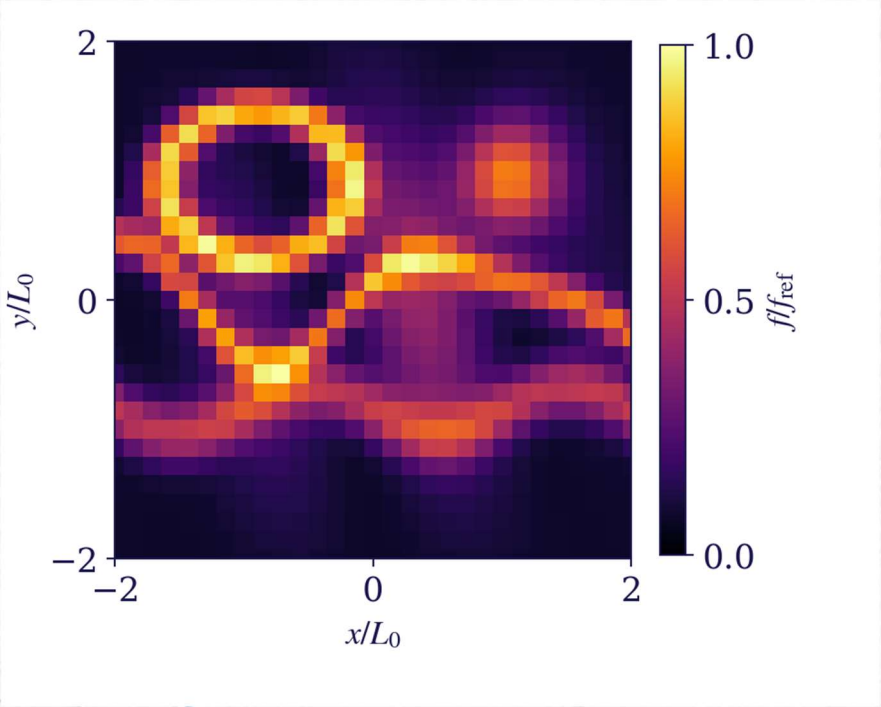
```
imagesc(x,y,0.1*Ei)
clim([0 1])
```

For quantitative comparisons, keep one mapping from value to colour.

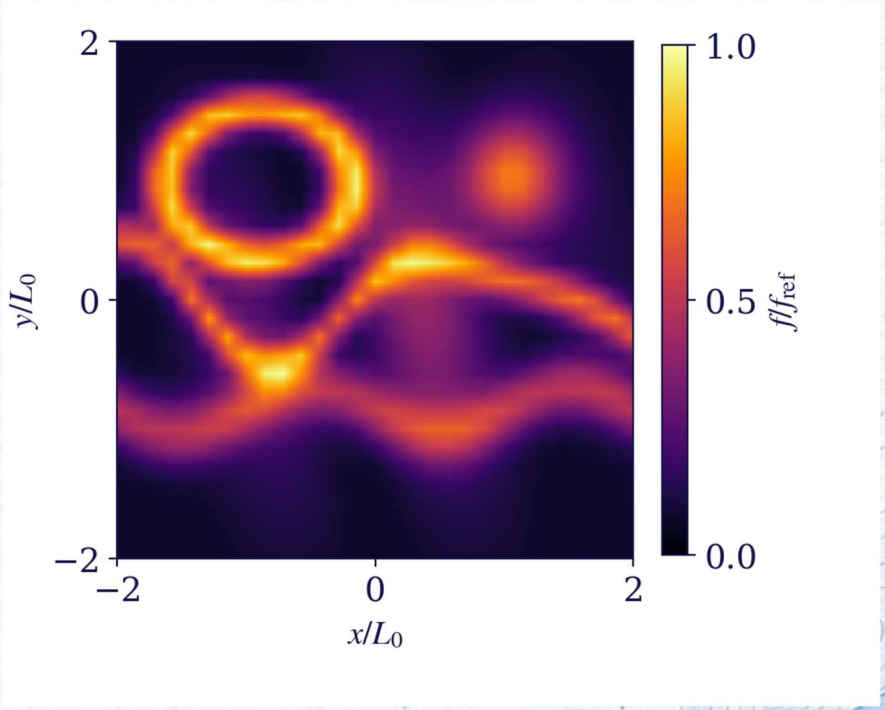
Interpolation and Display Sampling

Refine the display of a structured 2D field—not the original model grid.

29 × 29 original samples



401 × 401 interpolated display



MATLAB

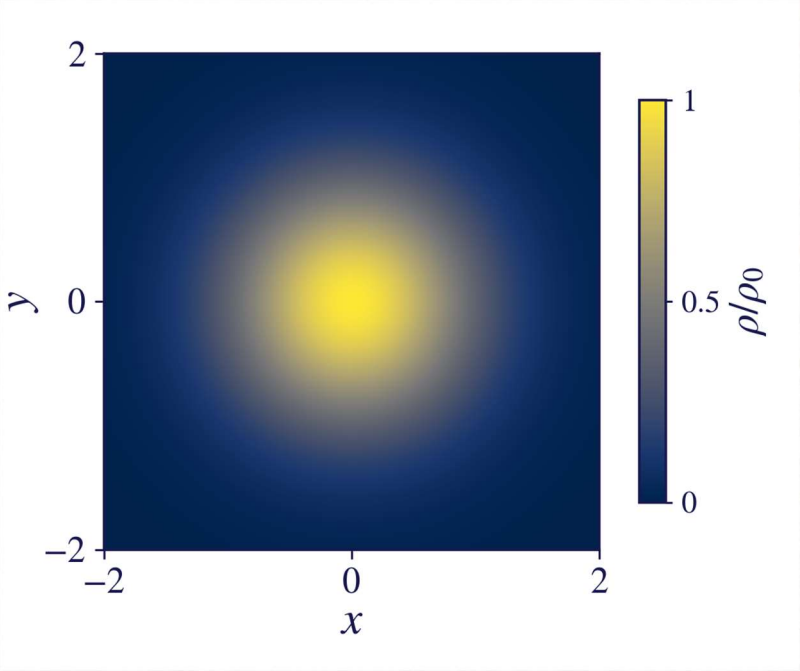
```
Fq = interp2(X,Y,F,Xq,Yq,'linear');
imagesc(xq,yq,Fq); axis xy image; clim([0 1]); colorbar
```

Same 841 samples and colour scale; a smoother image does not add model information.

Scalar Maps and Filled Contours

The same density array, represented by colour or selected levels.

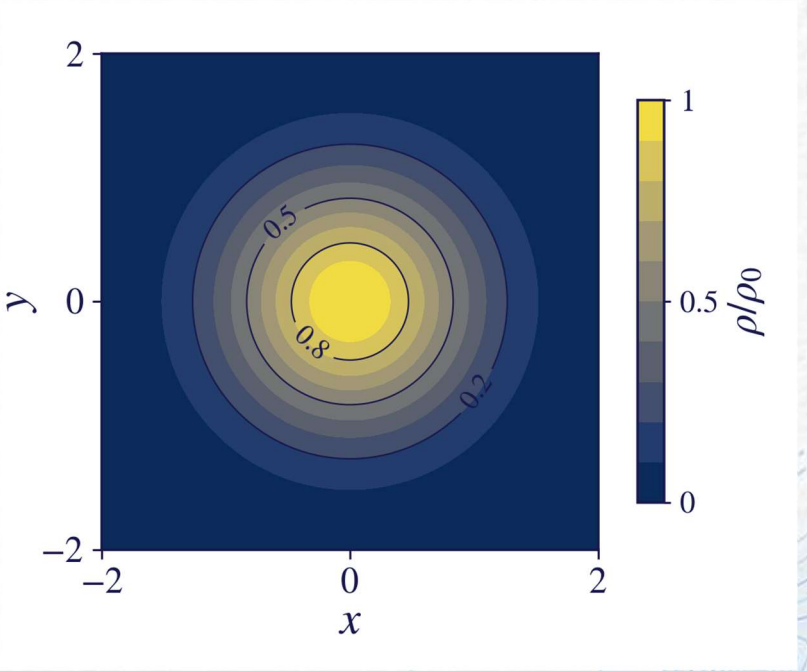
Colour map



MATLAB

```
imagesc(x,y,Rho)
axis xy image; colorbar
```

Filled contours



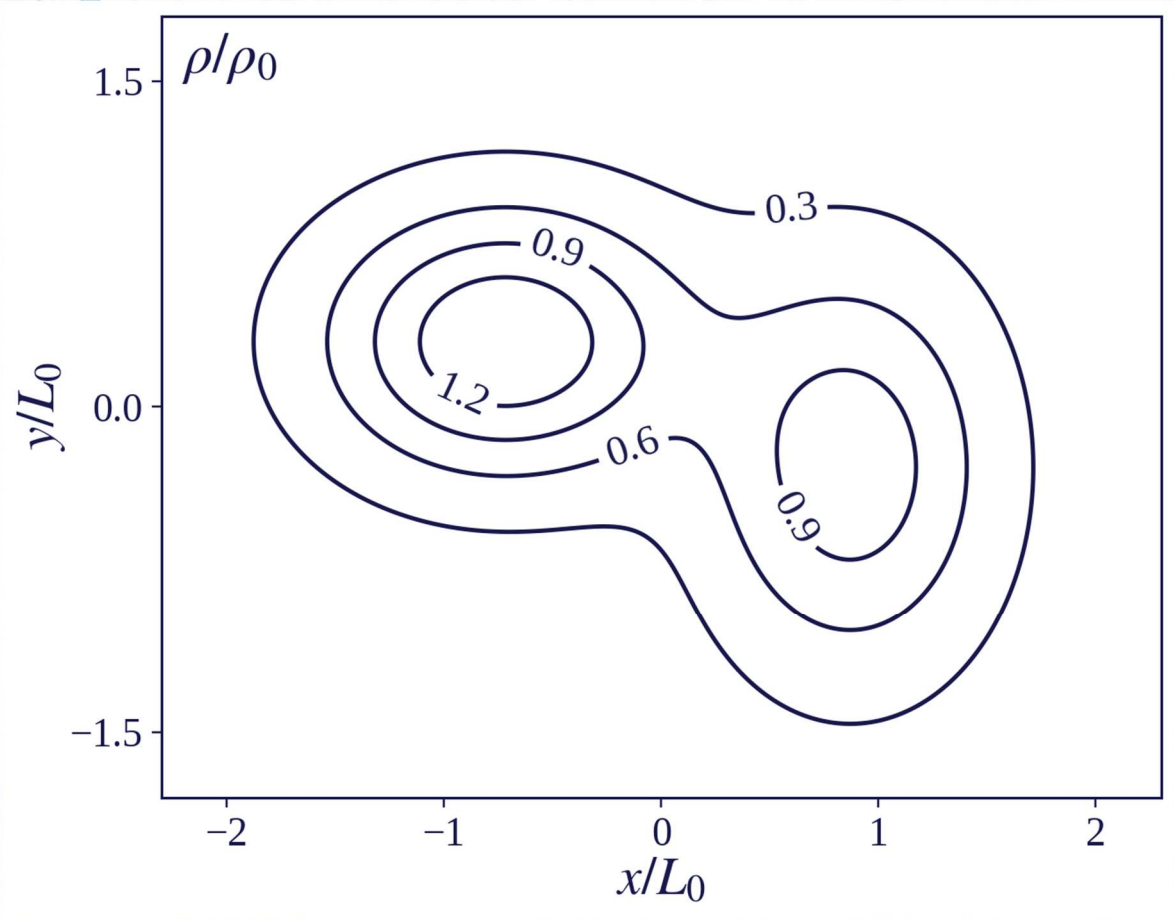
MATLAB

```
contourf(X,Y,Rho,0:.1:1)
axis equal tight; colorbar
```

Colour encodes values; contour boundaries mark selected values.

Contour Plots

Choose contour levels explicitly; label the values.



$$\rho(x, y) = \rho_i = \text{const.}$$

Equal level steps:
closer contours indicate
a steeper local gradient.

MATLAB

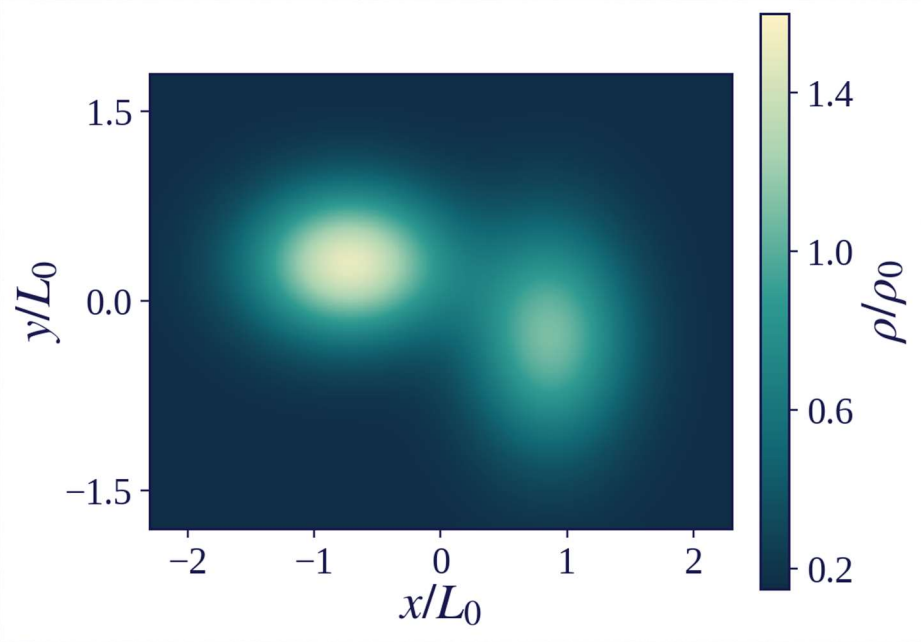
```
levels = [.3 .6 .9 1.2];
[C,h] = contour(X,Y,Rho,...
    levels);
clabel(C,h);
axis equal tight
```

Contour curves are scalar level sets—not generic vector field lines.

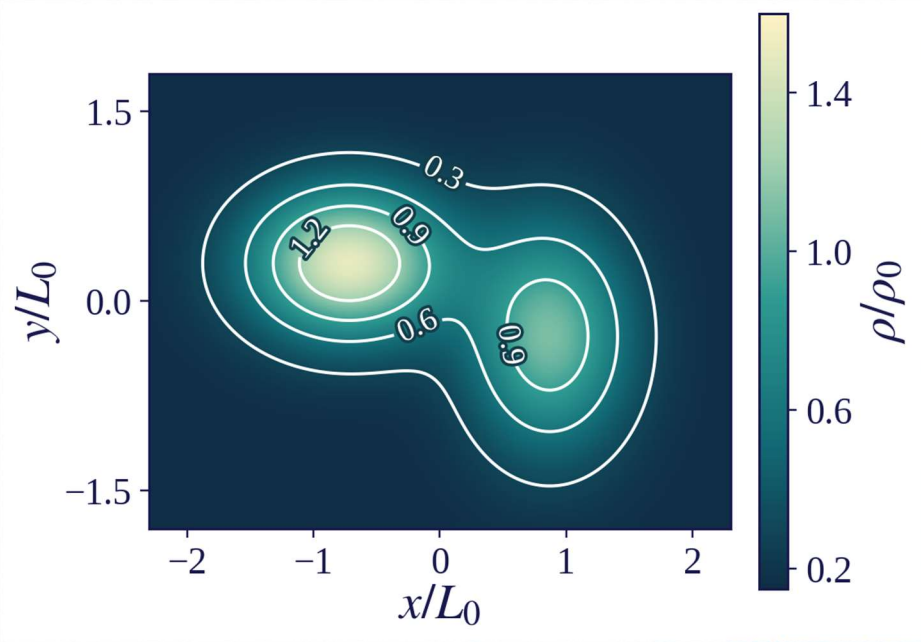
Density with Contours

Use hold on to combine registered layers in one coordinate system.

Density map



Density + contours



MATLAB

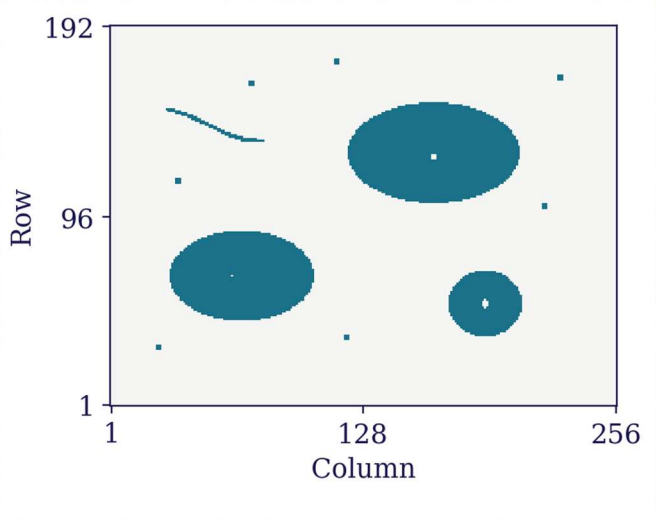
```
imagesc(x,y,Rho); axis xy image; hold on
[C,h] = contour(X,Y,Rho,levels,'w','LineWidth',1);
clabel(C,h,'Color','w'); hold off
```

Name each layer: here both colour and contours represent density.

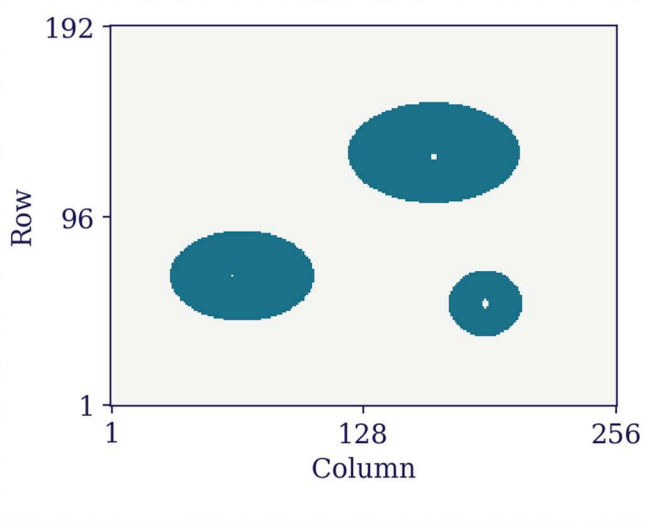
Morphological Filtering

Threshold a scalar field, then apply a declared structuring element.

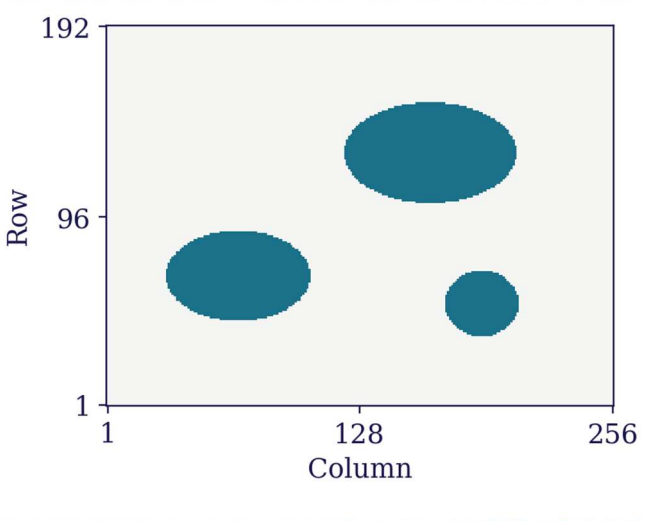
Threshold mask



Opening



Closing after opening



MATLAB

```
BW = F > 0.42;  se = strel('disk',3,0);
BWo = imopen(BW,se);  BWc = imclose(BWo,se);
imagesc(BWc); axis xy image; colormap gray
```

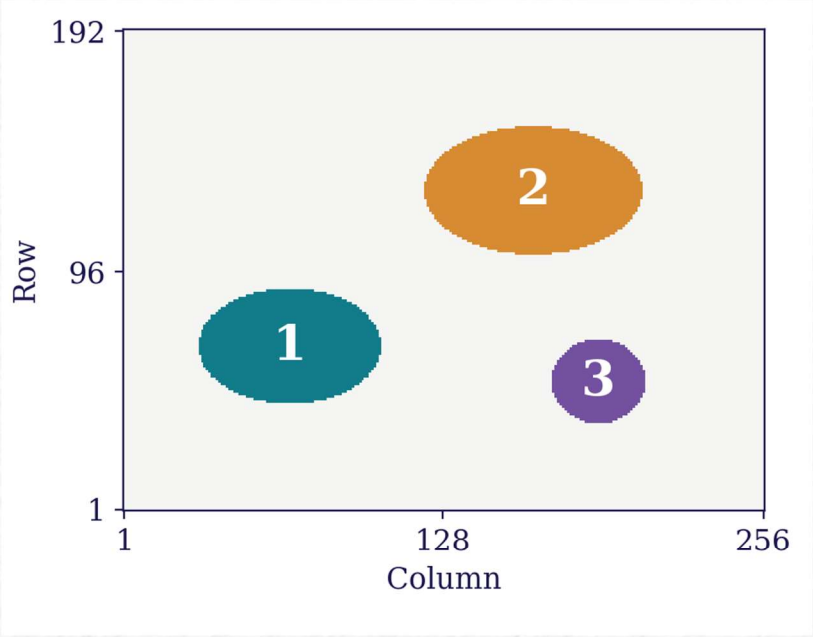
Opening removes small features; closing fills small gaps. The mask changes.

Disk radius: 3 pixels. Image Processing Toolbox. Analytic field definition in the notes.

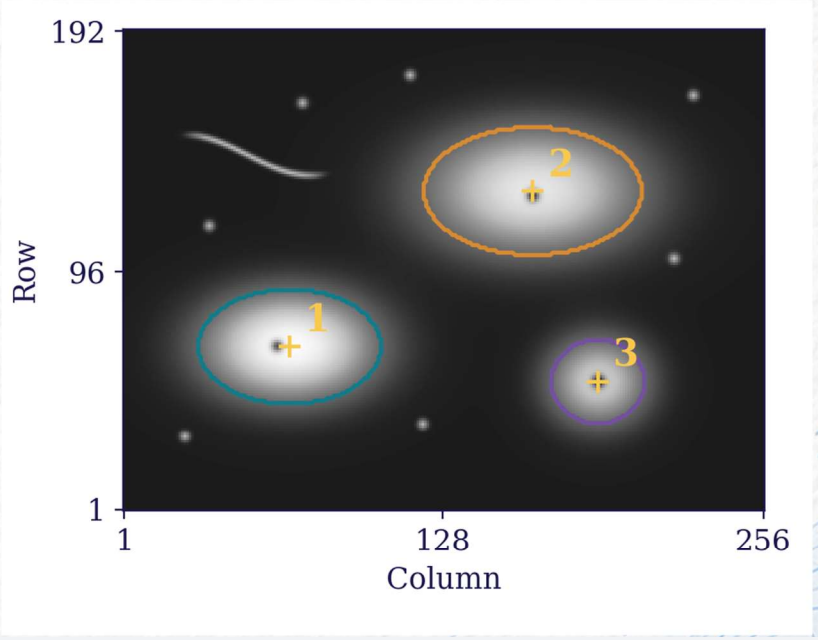
Morphology: Regions and Boundaries

Convert a mask into labelled objects and overlay their measured geometry.

Connected regions



Boundaries and centroids



MATLAB

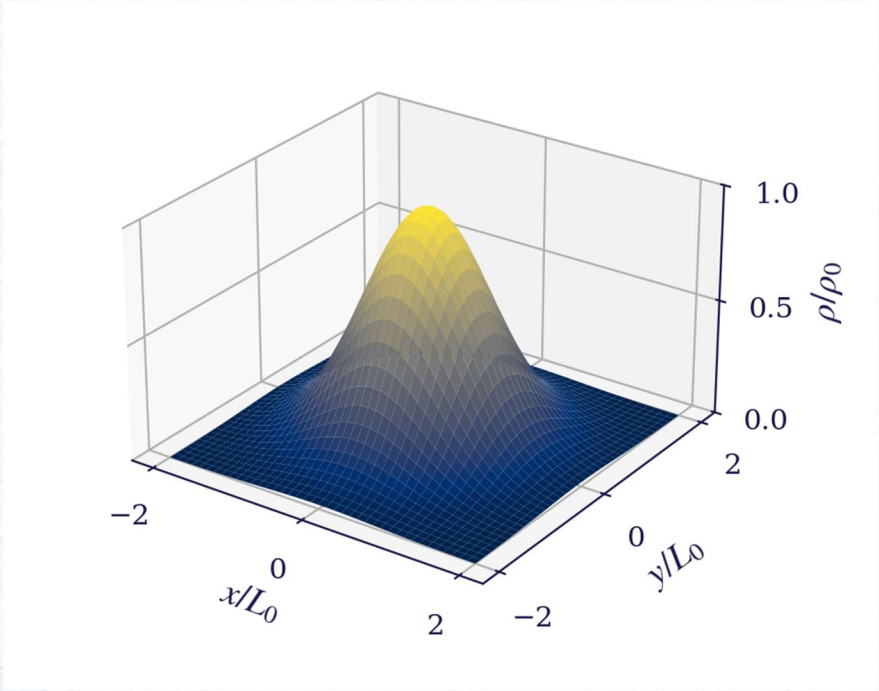
```
CC = bwconncomp(BWc,8); P = regionprops(CC,'Area','Centroid');
imagesc(F); axis xy image; hold on
visboundaries(BWc,'Color','c'); c = vertcat(P.Centroid);
plot(c(:,1),c(:,2),'y+');
```

Labels identify objects; area counts pixels. Report threshold and connectivity.

Surface and Mesh Plots

Use height for the scalar value; keep the coordinate meaning explicit.

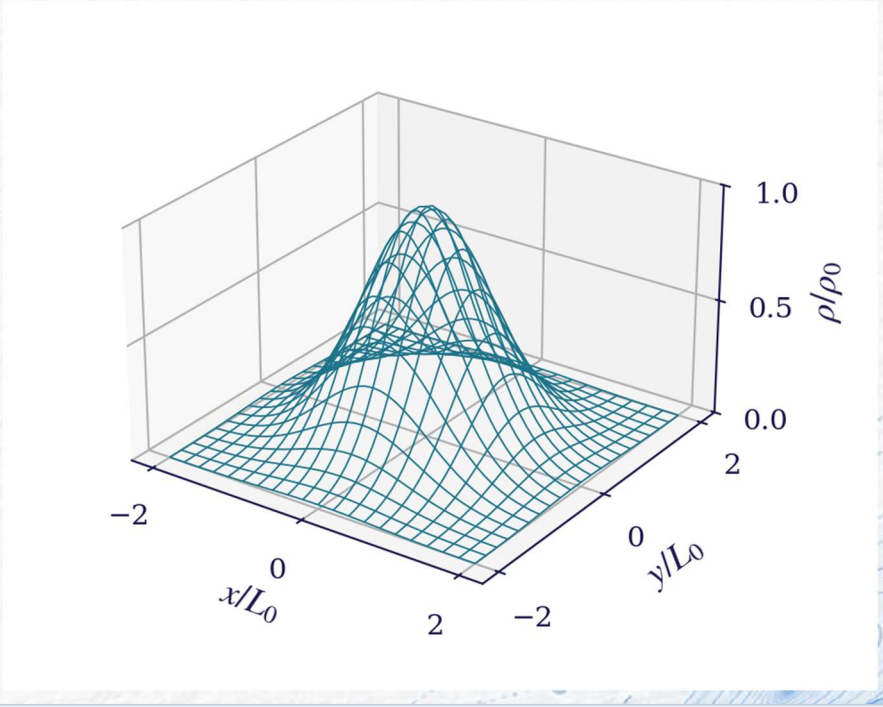
Filled surface



MATLAB

```
surf(X,Y,Rho)
shading interp; view(3)
```

Wireframe mesh



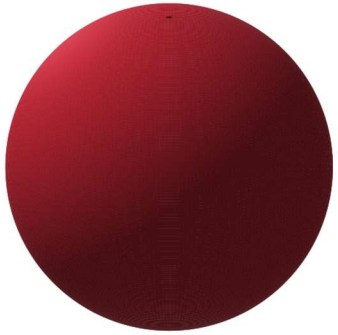
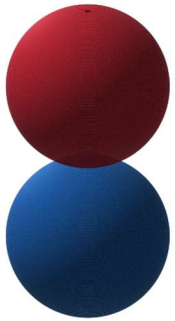
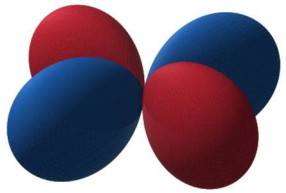
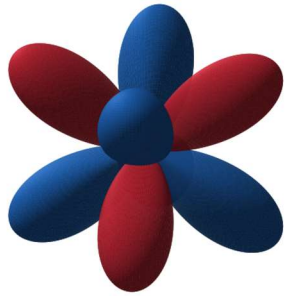
MATLAB

```
mesh(X,Y,Rho)
view(3); grid on
```

The plotting height is ρ/ρ_0—not a third physical coordinate.

Multipole Surfaces: d = 0–3

Use radial distance for angular amplitude and colour for its sign.

Monopole d = 0	Dipole d = 1	Quadrupole d = 2	Octupole d = 3
			
1	n_z	$n_x^2 - n_y^2$	$2\, n_x\, n_y\, n_z$

$$r = |F_d(\theta, \phi)|, \qquad \mathbf{x} = r\,\hat{\mathbf{n}}, \qquad \max|F_d| = 1$$

MATLAB

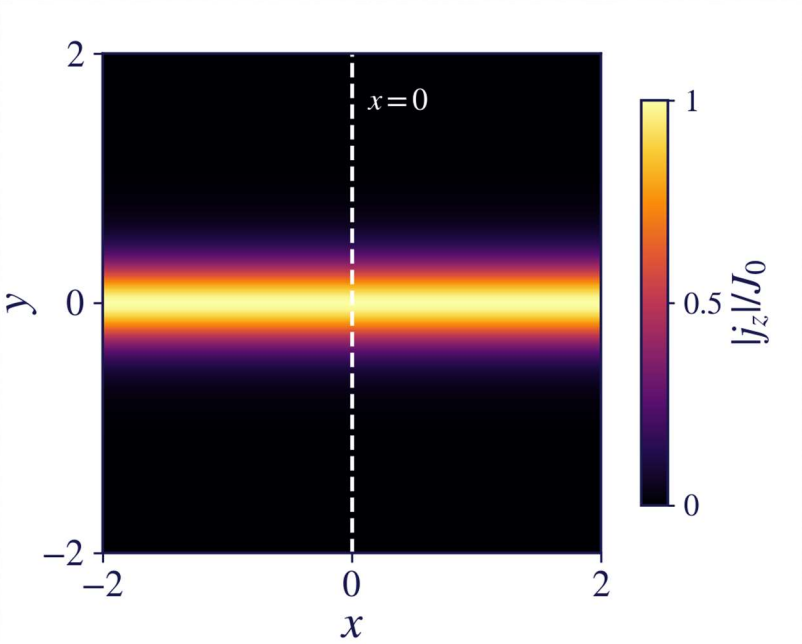
```
[nx,ny,nz] = sphere(160);
A = {ones(size(nx)), nz, nx.^2-ny.^2, 2*nx.*ny.*nz};
d = 3; F = A{d+1}; F = F/max(abs(F(:))); r = abs(F);
surf(r.*nx,r.*ny,r.*nz,F,'EdgeColor','none'); axis equal off
colormap([.12 .38 .70; .77 .15 .21]); clim([-1 1]); view(3)
```

Selected real spherical harmonics. Red: positive; blue: negative. Surfaces are amplitude glyphs.

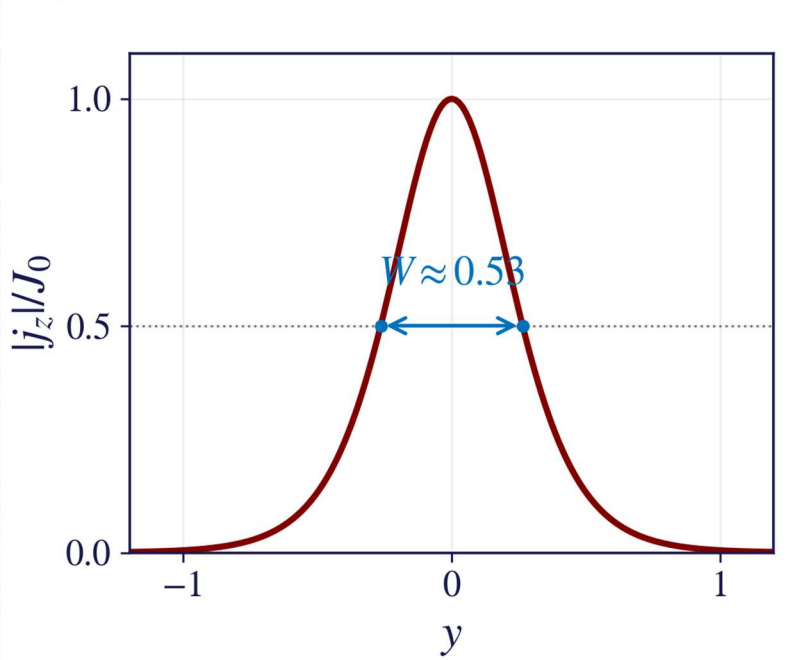
Extracting a Line Profile

Show the selected cut beside the values extracted from the array.

Current-density map



Cross-sheet profile



MATLAB

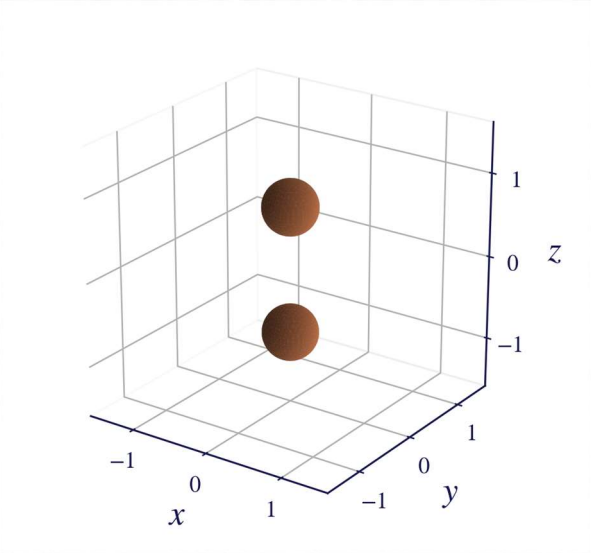
```
[~,i0] = min(abs(x));    % column nearest x = 0
profile = Jabs(:,i0);
plot(y,profile,'LineWidth',1.5); grid on
```

Plot numerical samples—not colour values read from a screenshot.

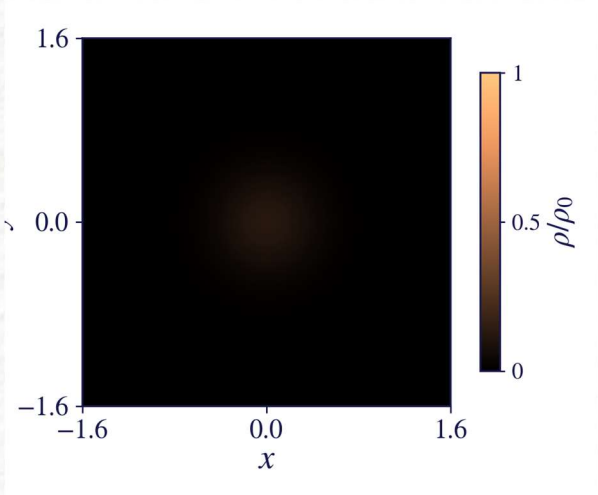
Slices Through a Volume

Rho3(row,column,page) stores a scalar on a three-dimensional grid.

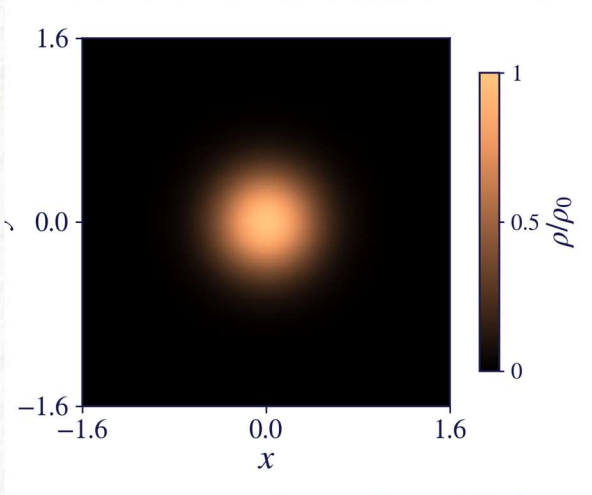
3D density



Slice: z = 0



Slice: z = 0.75



MATLAB

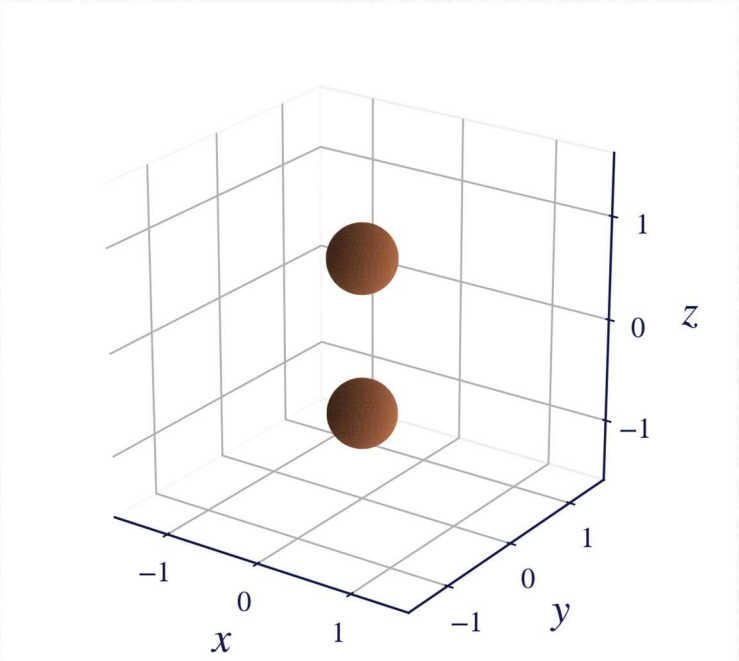
```
h = slice(X,Y,Z,Rho3,[],[],[0 .75]);
set(h,'EdgeColor','none'); daspect([1 1 1]);
view(3); colorbar; clim([0 1])
```

slice interpolates at the requested plane; it does not integrate through the volume.

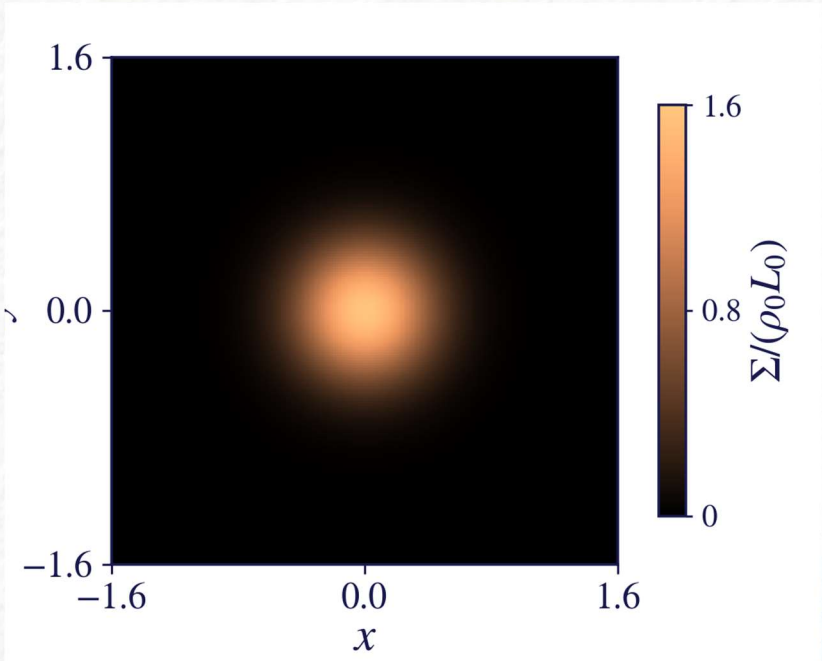
Projection by Integration

Reduce a volume to a two-dimensional array before plotting.

Volume



Column-density map



MATLAB

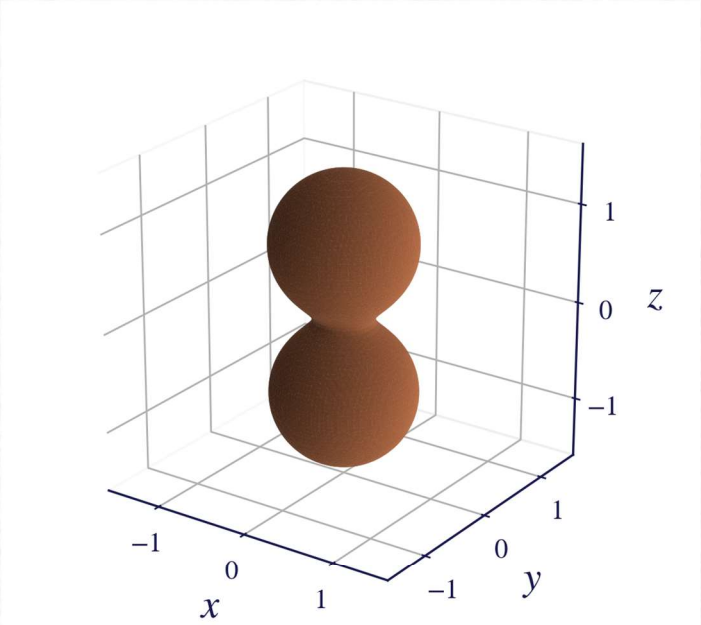
```
Sigma = trapz(z,Rho3,3);   % integrate the third dimension
imagesc(x,y,Sigma); axis xy image; colorbar
```

A projection sums contributions along a direction; a slice selects a plane.

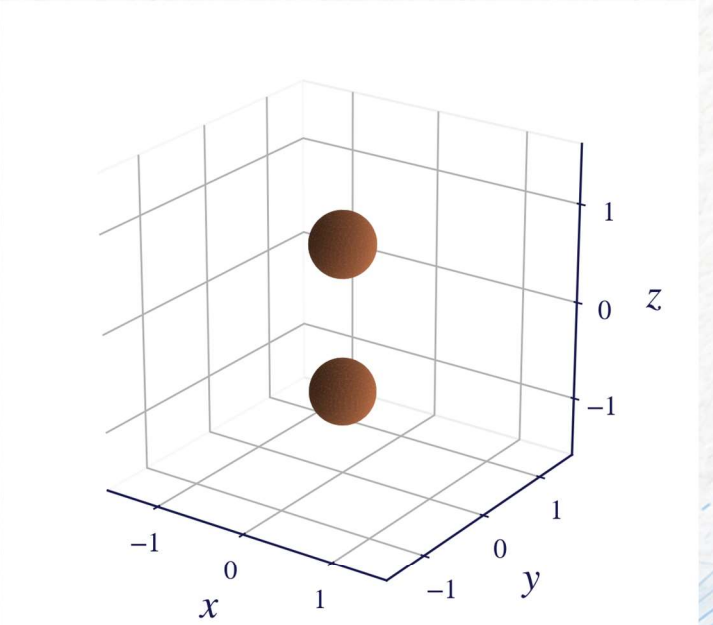
Isosurfaces

Display one scalar level as a surface in three-dimensional space.

Low density level: 0.08



High density level: 0.60



MATLAB

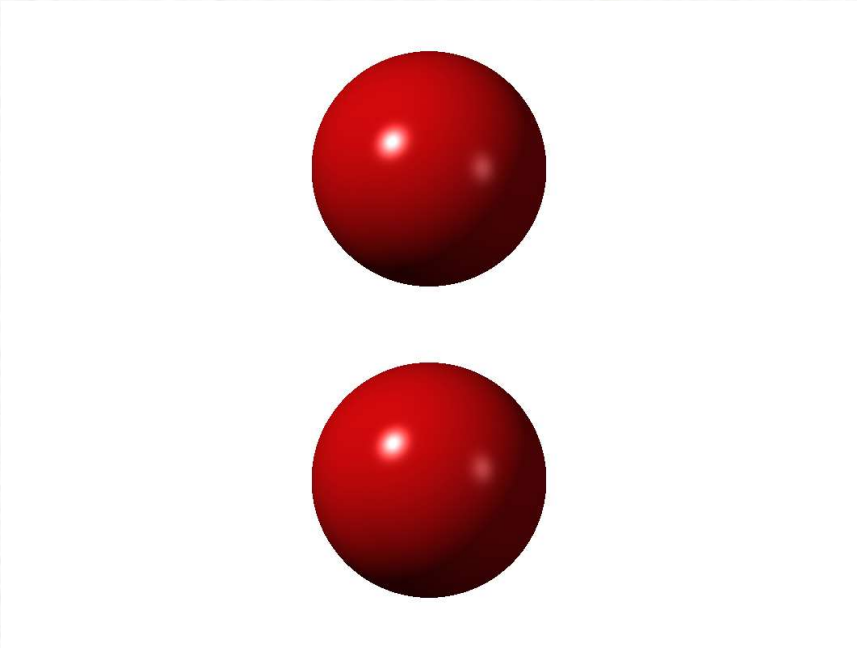
```
p = patch(isosurface(X,Y,Z,Rho3,level));
isonormals(X,Y,Z,Rho3,p);
set(p,'FaceColor',[.74 .44 .28],'EdgeColor','none');
daspect([1 1 1]); view(3); camlight; lighting gouraud
```

The chosen level defines the displayed surface; it is not a material boundary.

Camera, Aspect and Lighting

Two larger red spheres; change the view, not the geometry.

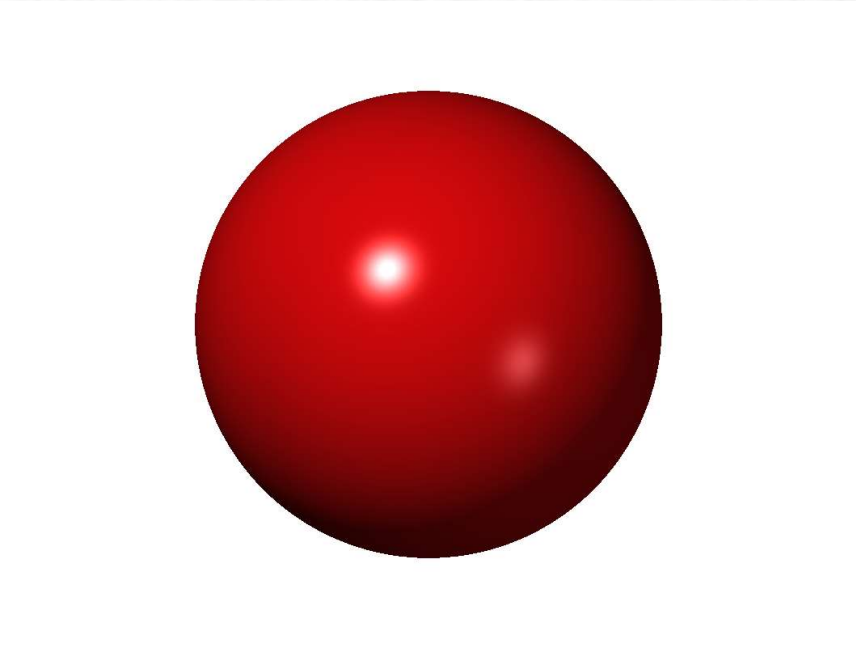
- Oblique view: two spheres



MATLAB

```
set(p,'FaceColor',[.83 .04 .05]);
axis equal off; view(3)
camlight(-30,40); lighting gouraud
material shiny
```

- Along z: the front sphere hides the rear



MATLAB

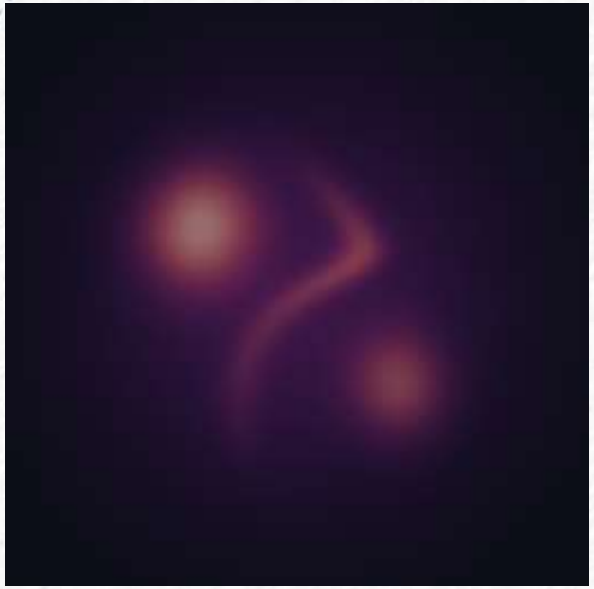
```
set(p,'FaceColor',[.83 .04 .05]);
axis equal off; view(2)
camlight(-30,40); lighting gouraud
material shiny
```

Bright highlights and dark limbs show illumination—not a change of the scalar field.

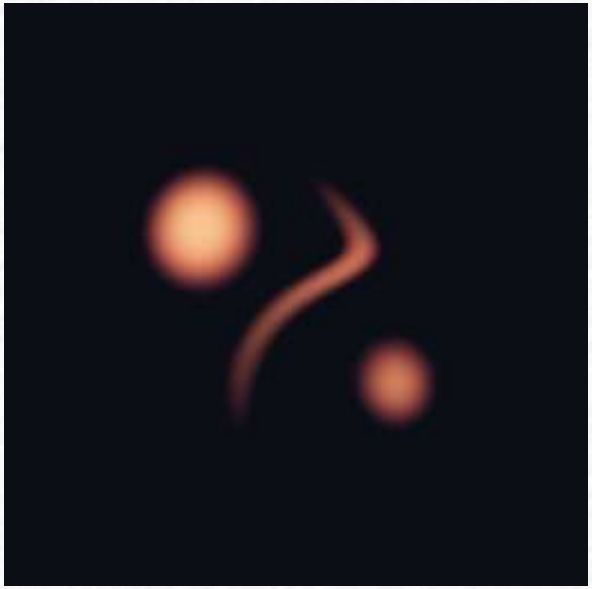
Volume Rendering and Opacity

Values throughout a volume contribute through colour and opacity.

Diffuse material

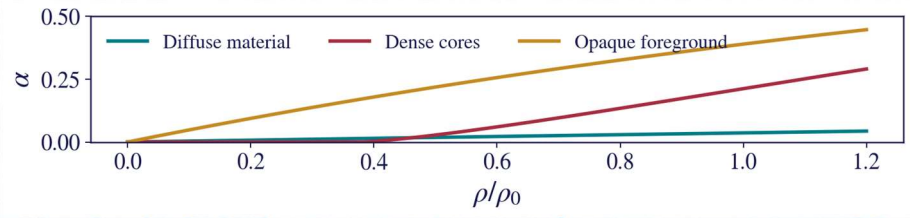


Dense cores



Opaque foreground





MATLAB

```
v = volshow(V, ...
    Colormap=cm,Alphamap=alpha);
% Image Processing Toolbox
```

Hold the camera and colours fixed when comparing opacity transfer functions.

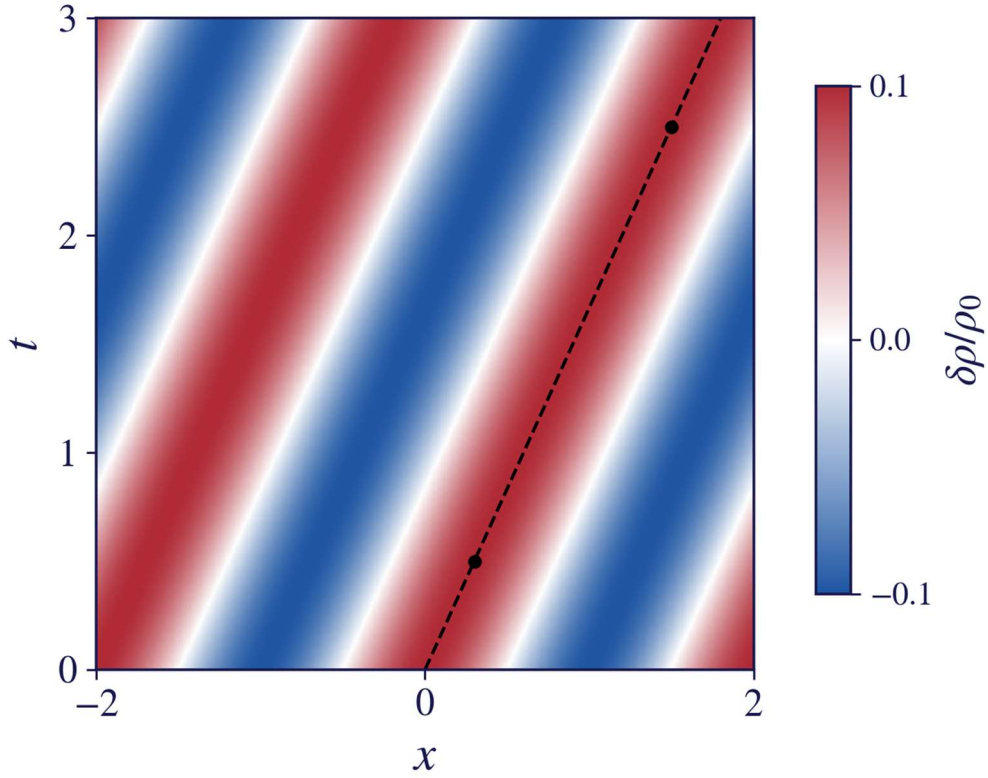
Space–Time Maps

Stack one-dimensional field samples into a two-dimensional image.

Columns: position x
Rows: time t
Colour: $\delta\rho/\rho_0$

MATLAB

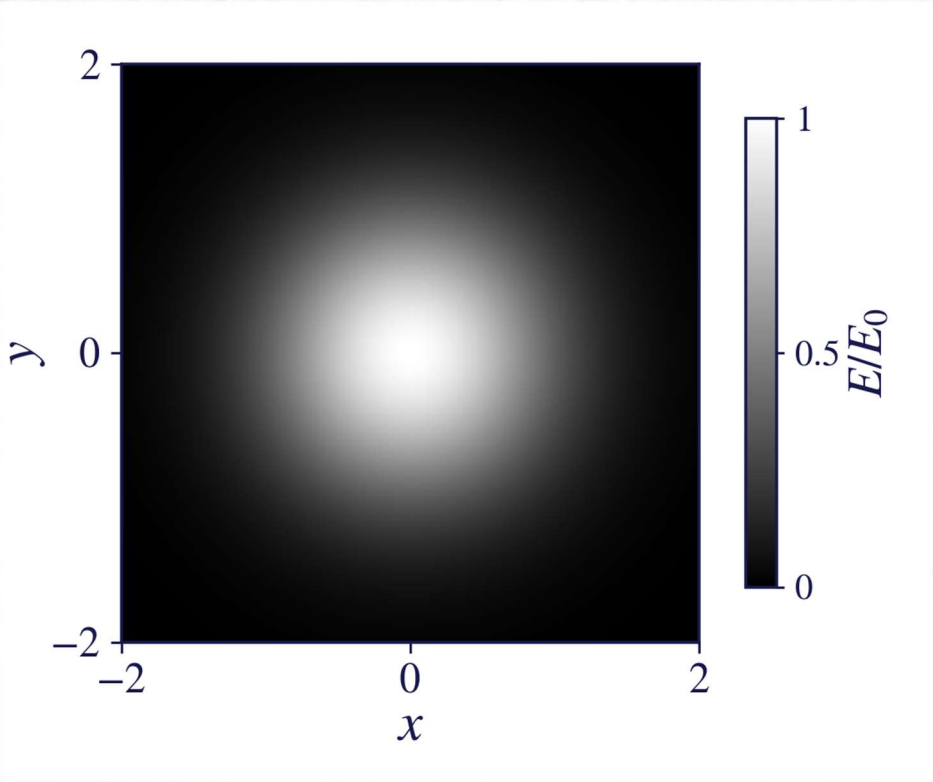
```
for k = 1:numel(t)
    F(k,:) = .1*cos( ...
        pi*(x-.6*t(k)));
end
imagesc(x,t,F); axis xy
clim([-.1 .1]); colorbar
```



The second image coordinate is time—not a second spatial dimension.

Animating a Field

Create the image once; update its CData for each model time.



MATLAB

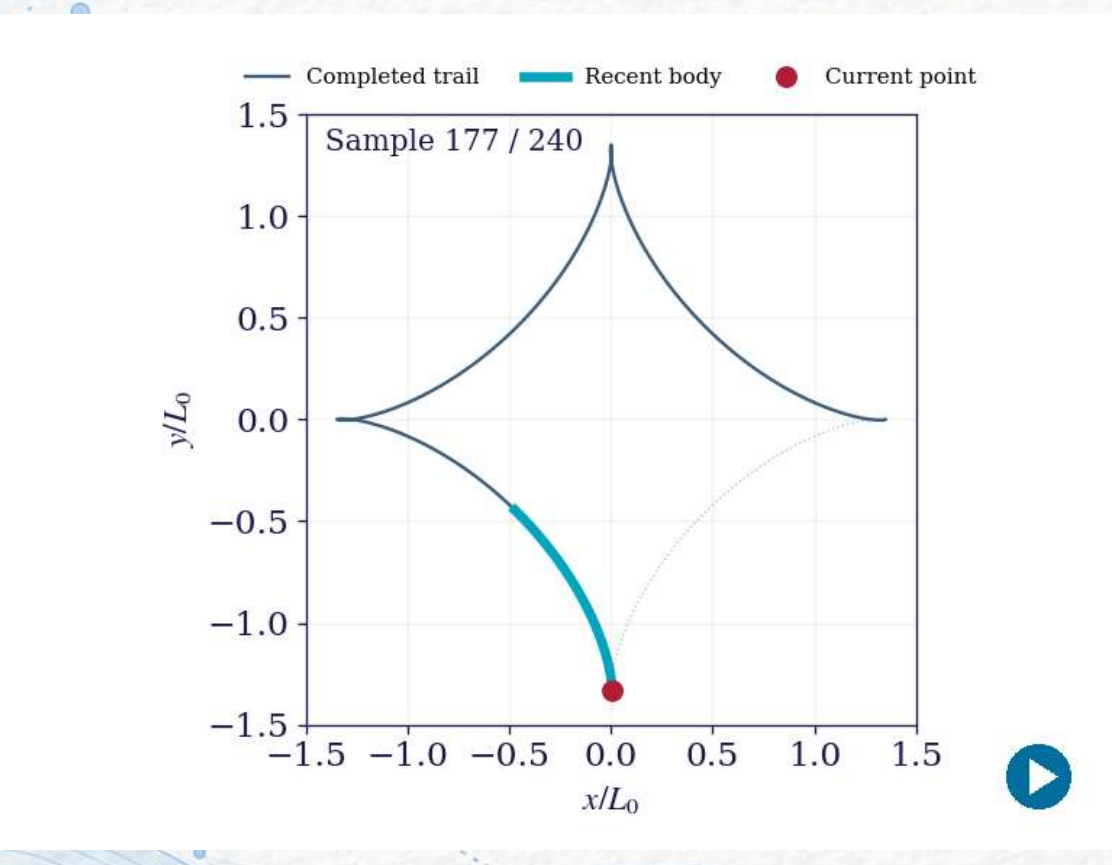
```
h = imagesc(x,y,Ei,[0 1]);
axis xy image; colormap gray
for t = 0:.05:3
    h.CData = exp(-t)*Ei;
    title(sprintf('t = %.2f',t))
    drawnow
end
```

Keep axes, colour limits and camera fixed.

Animation changes the displayed array; plotting settings should not drift.

Animation: comet

Animate a moving point, a recent body segment and a completed trail.



Head → current sample
Body → recent samples
Trail → completed path

MATLAB

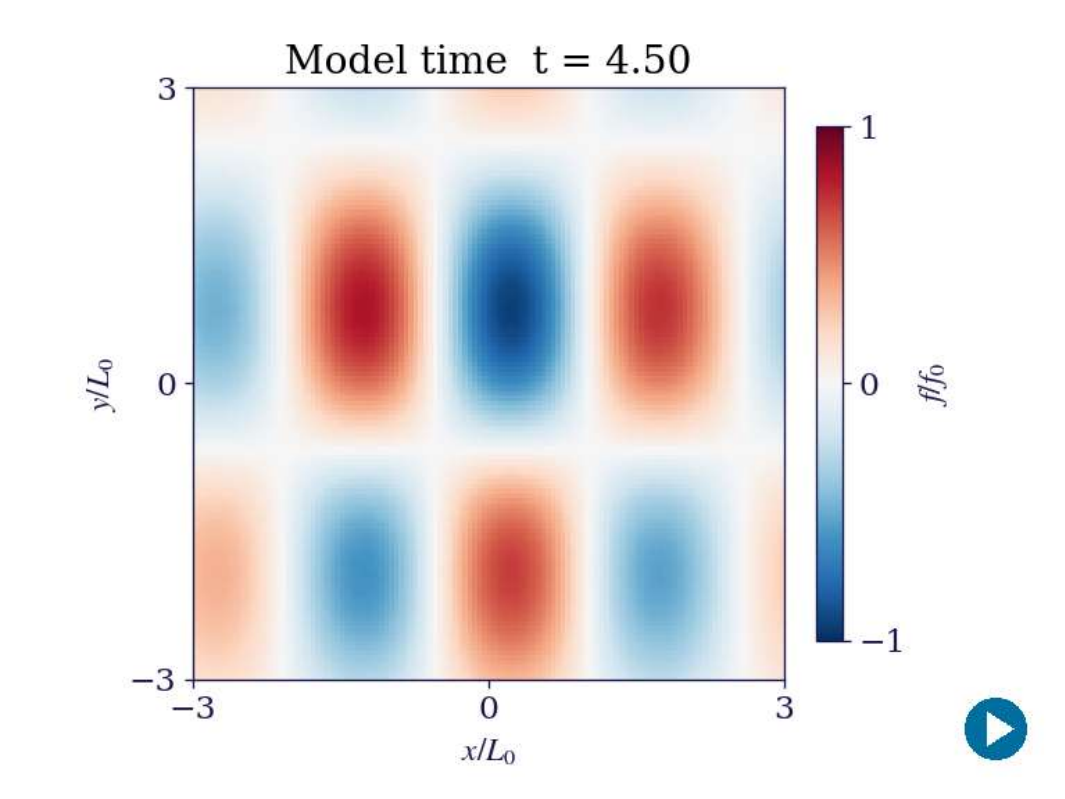
```
t = linspace(0,2*pi,240);
xp = cos(t)+.35*cos(3*t);
yp = sin(t)-.35*sin(3*t);
comet(xp,yp,.12);
axis equal
```

p = 0.12 sets the body length as a fraction of the samples. Click the plot to play.

A comet plot animates a sampled curve; its playback speed is not a model time step.

Animation: Save a Video File

Update the field, capture each figure, and write frames at a fixed playback rate.



MATLAB

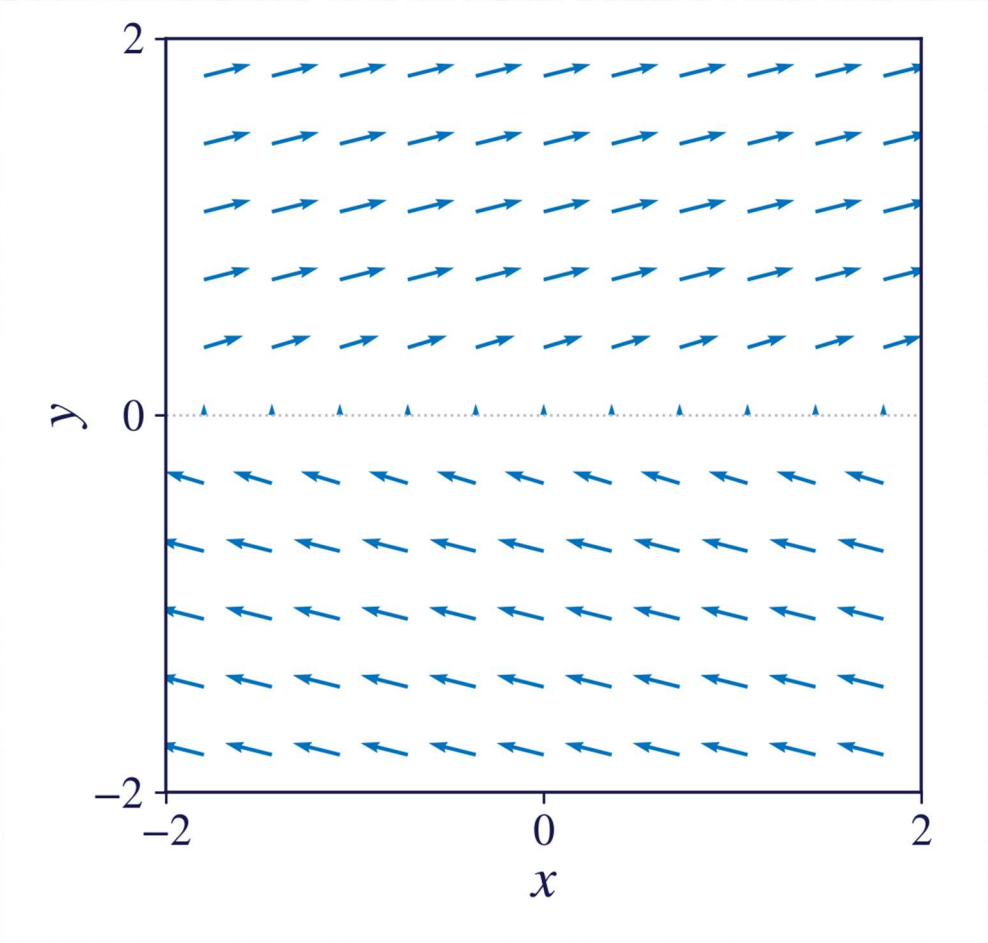
```
v = VideoWriter('field.mp4','MPEG-4');
v.FrameRate = 24; open(v);
h = imagesc(x,y,F(:,:,1),[-1 1]);
axis xy image; colorbar
for k = 1:size(F,3)
    h.CData = F(:,:,k);
    drawnow
    writeVideo(v,getframe(gcf));
end
close(v)
```

192 frames / 24 fps = 8 s
F(:,:,k) is evaluated analytically.
The complete formula is in the notes.

FrameRate sets playback; the model time array sets the physical chronology.

Arrow Plots

Sample a vector field with glyphs placed at known coordinates.



$$\mathbf{B}/B_0 = (\tanh(y/a),\ 0.25)$$

Arrow direction → B
Arrow length → |B|
Use a sparse sample.

MATLAB

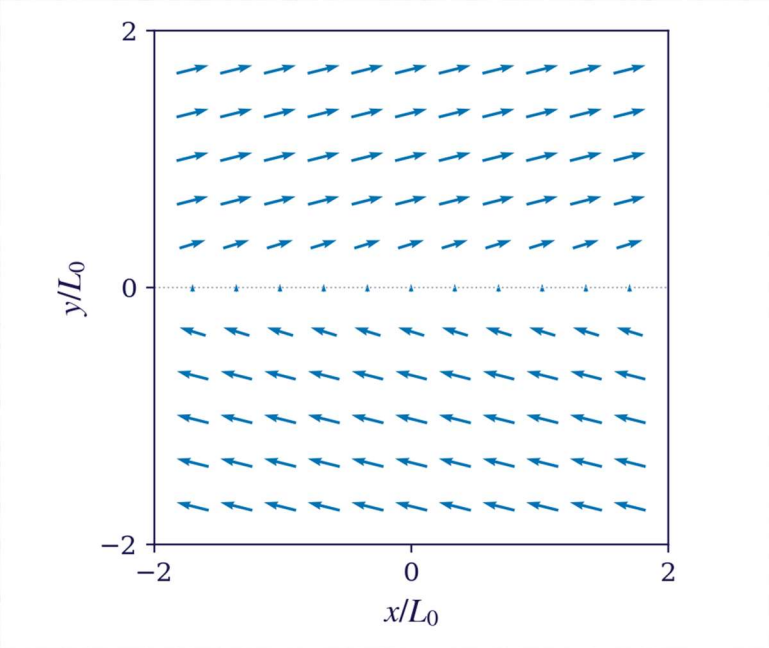
```
ii = 1:10:numel(x);
quiver(X(ii,ii),Y(ii,ii),...
    Bx(ii,ii),By(ii,ii));
axis equal tight
```

Automatic arrow scaling is a display factor; use one scale for comparisons.

Direction versus Magnitude

Decide whether arrow length should carry information.

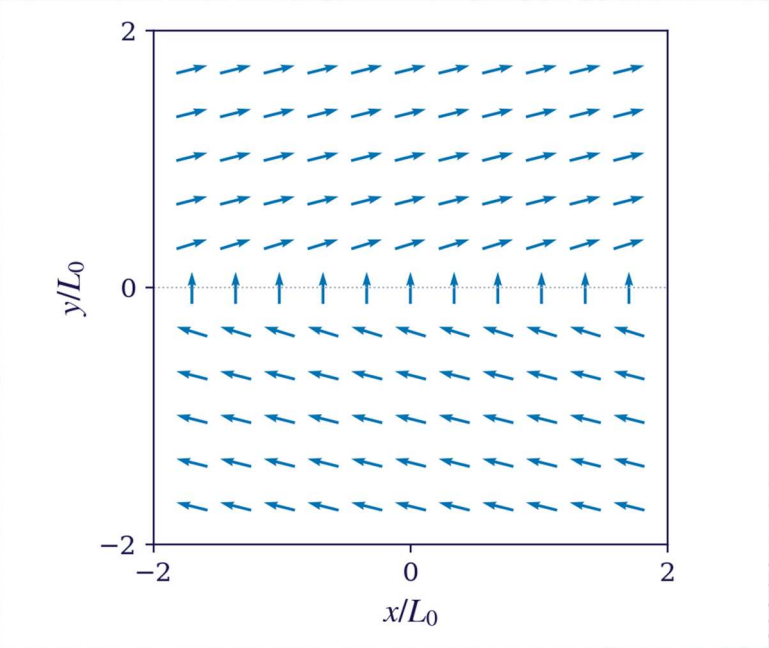
Length proportional to |B|



MATLAB

```
quiver(X,Y,Bx,By)
axis equal tight
```

Equal-length direction arrows



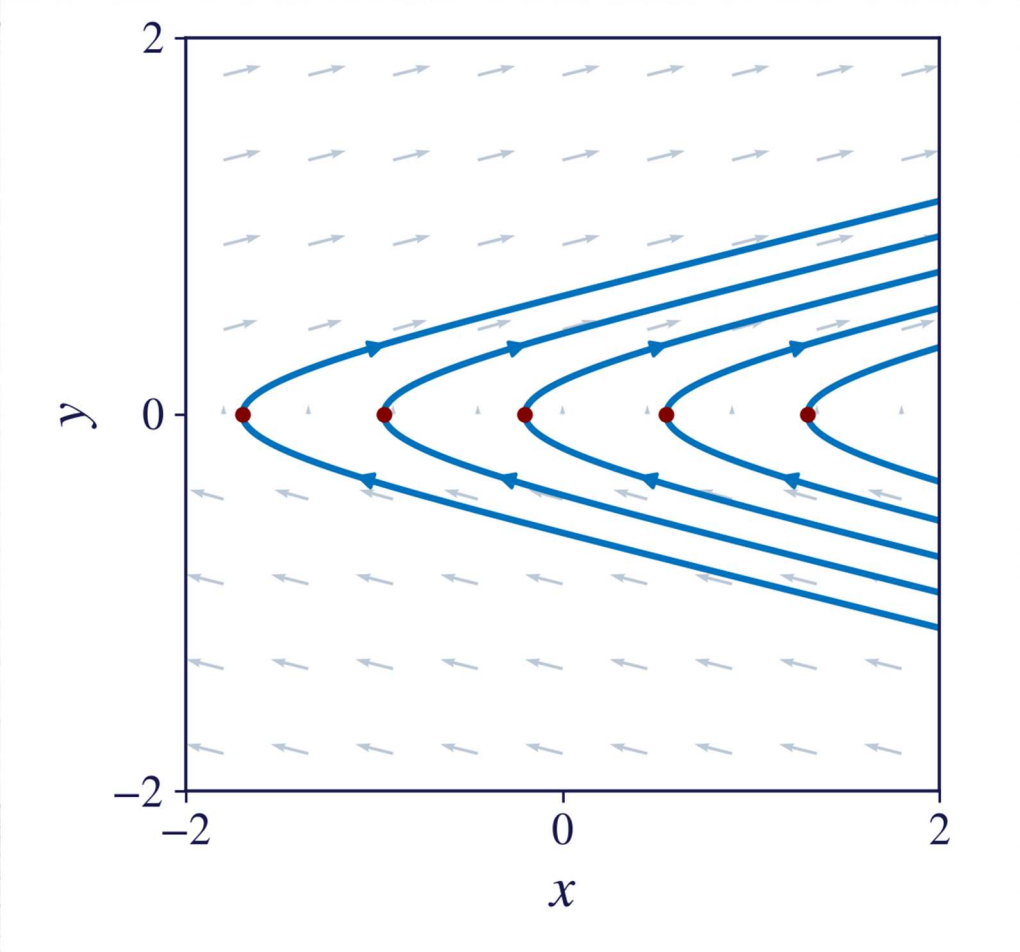
MATLAB

```
b = hypot(Bx,By); b(b==0)=NaN;
quiver(X,Y,Bx./b,By./b)
axis equal tight
```

Normalizing arrows removes magnitude information; mask zero vectors.

Field Lines and Seed Points

Integrate curves tangent to the vector field at one fixed time.



Choose seed locations.
Trace both directions.
Avoid crowded curves.

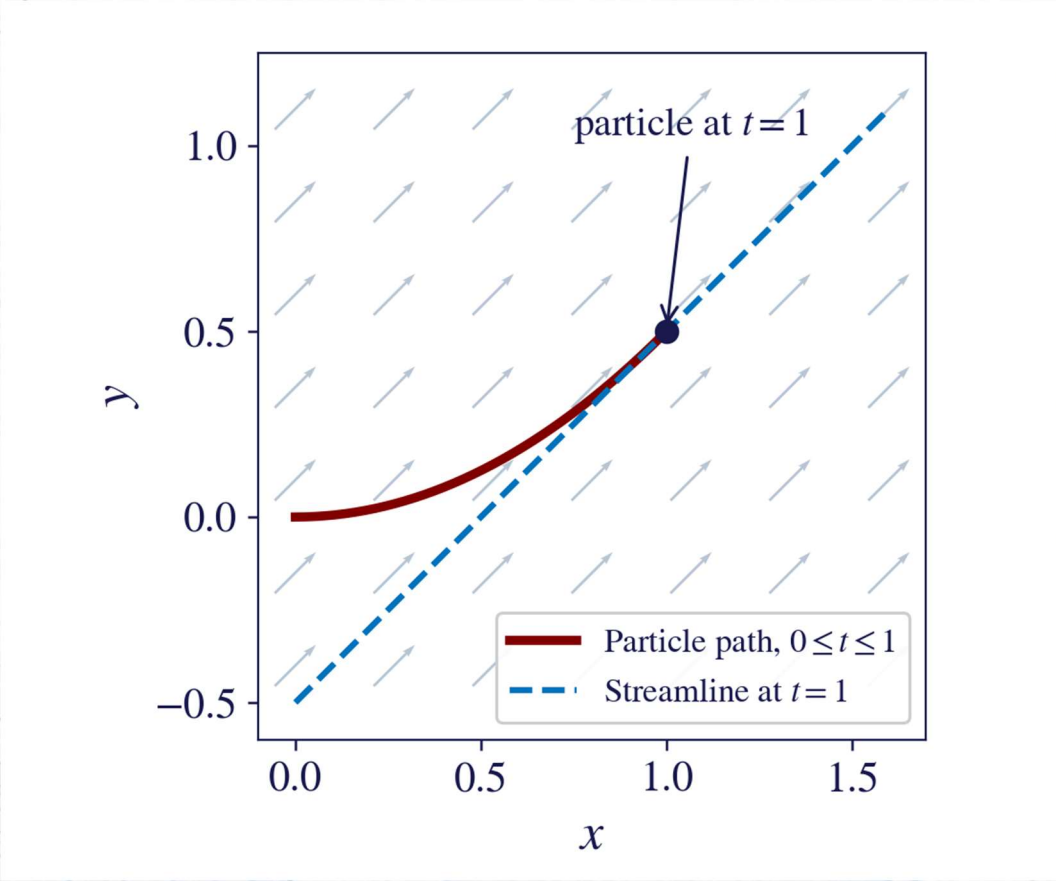
MATLAB

```
sx = [-1.5 -.8 -.1 .6];
sy = zeros(size(sx));
streamline(X,Y,Bx,By,sx,sy);
hold on
streamline(X,Y,-Bx,-By,sx,sy);
axis equal tight
```

Adding seeds adds drawn curves—not field strength or magnetic flux.

Streamlines and Trajectories

Freeze the field for streamlines; integrate time for trajectories.



$$\mathbf{v}(x, y, t) = (1, t)$$

MATLAB

```
% trajectory: v = (1,t)
[t,r] = ode45(@(t,r)[1;t],...
    [0 1],[0;0]);
plot(r(:,1),r(:,2));
hold on
% streamline at t = 1
plot(x,x-.5,'--'); axis equal
```

The same plotting tools display different curves; state which curve is shown.

Analytic Construction: Magnetic Islands

Create coordinates and field arrays directly—no input data file.

$$D=\cosh y+\varepsilon\cos x,\quad A_z=\ln D,\quad \mathbf{B}=\left(\frac{\sinh y}{D},\frac{\varepsilon\sin x}{D},0\right)$$

$$j_z=-\frac{1-\varepsilon^2}{D^2},\qquad \varepsilon=0.35,\qquad B_0=a=\mu_0=1$$

MATLAB

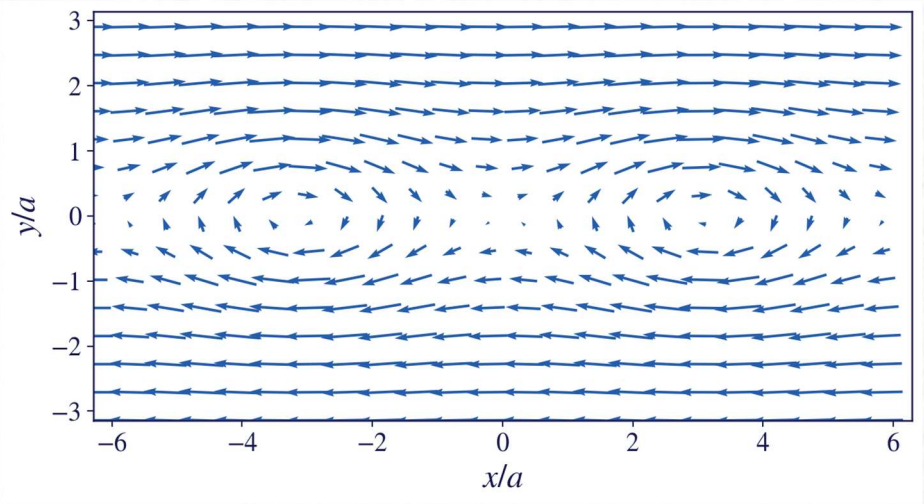
```
N=161; epsilon=0.35; a=1; B0=1; mu0=1;
x=linspace(-2*pi,2*pi,N);
y=linspace(-pi,pi,(N+1)/2); [X,Y]=meshgrid(x,y);
D=cosh(Y)+epsilon*cos(X); Az=log(D);
Bx=sinh(Y)./D; By=epsilon*sin(X)./D;
Jz=-(1-epsilon^2)./D.^2; Jz_exact=Jz;
AX=log(1+epsilon); AO=log(1-epsilon);
```

Rows follow y; columns follow x. The same arrays support arrows, flux contours and LIC.

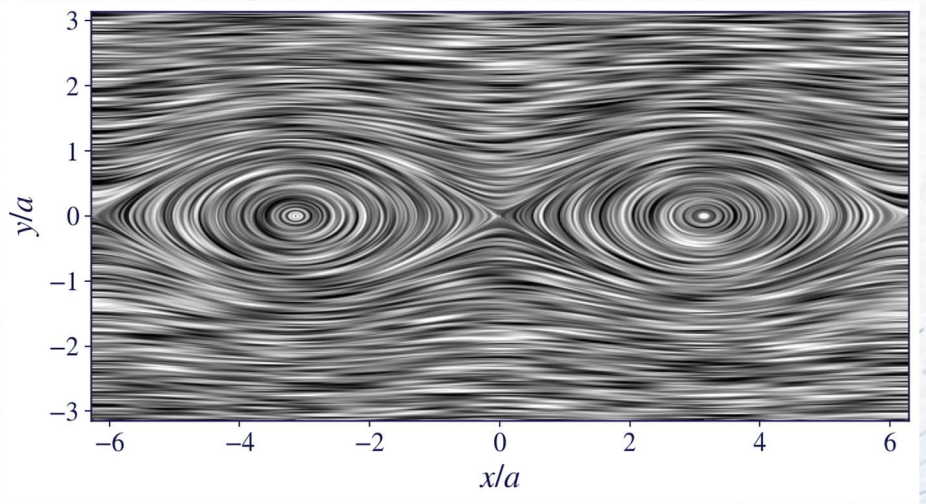
Line Integral Convolution

Filter a noise texture along local vector-field directions.

Sparse arrows



Dense LIC texture



MATLAB

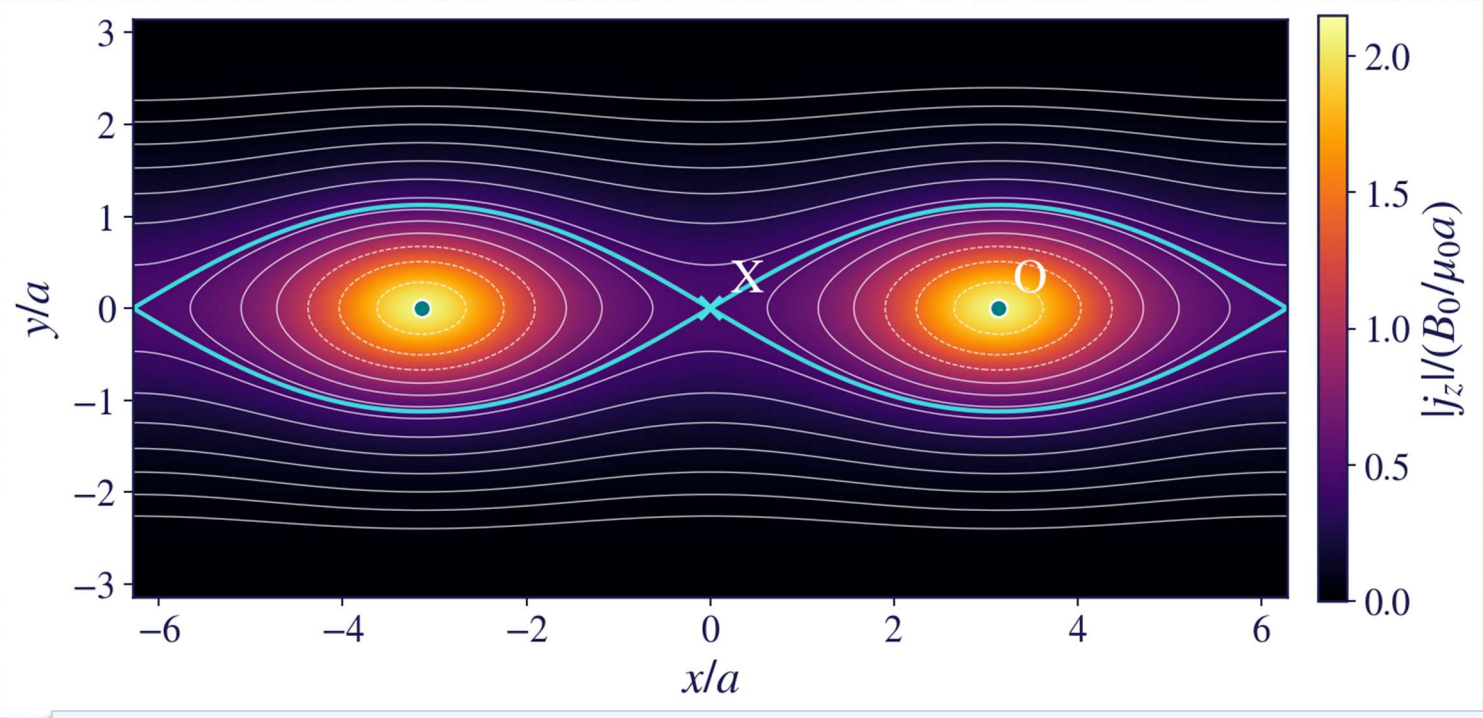
```
L = lic2(Bx,By,40);    % supplied helper; equal grid spacing
imagesc(x,y,L); axis xy image; colormap gray
```

lic2.m is included with the MATLAB examples; it is not a built-in function.

Symmetric LIC encodes orientation—not vector sign or calibrated magnitude.

Flux Contours as Field Lines

In this 2D model, flux contours trace the in-plane magnetic field.



Contours: flux
Colour: |j_z|
Cyan: separatrix

MATLAB

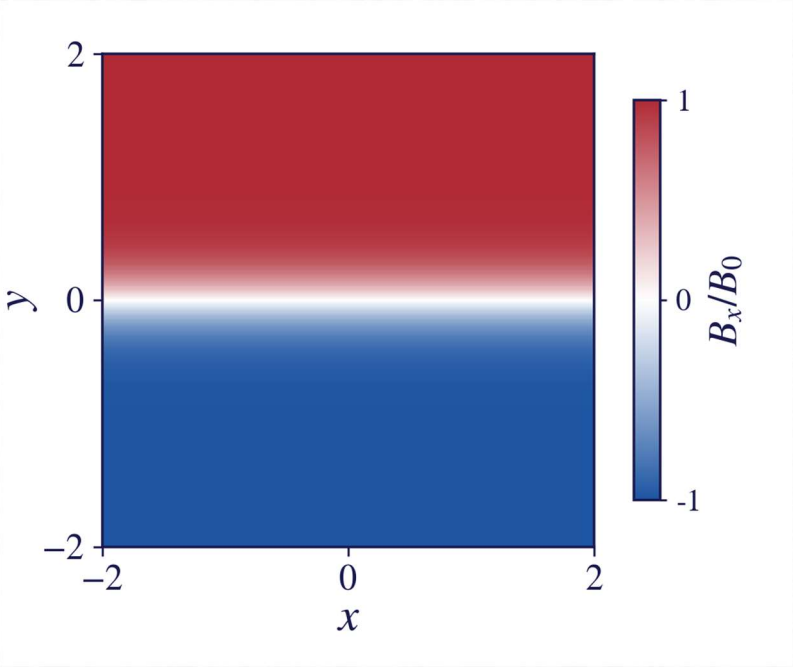
```
imagesc(x,y,abs(Jz)); axis xy image; hold on
contour(X,Y,Az,25,'w');
contour(X,Y,Az,[AX AX],'c','LineWidth',1.5)
```

A scalar contour is a field line only when the defining relation justifies it.

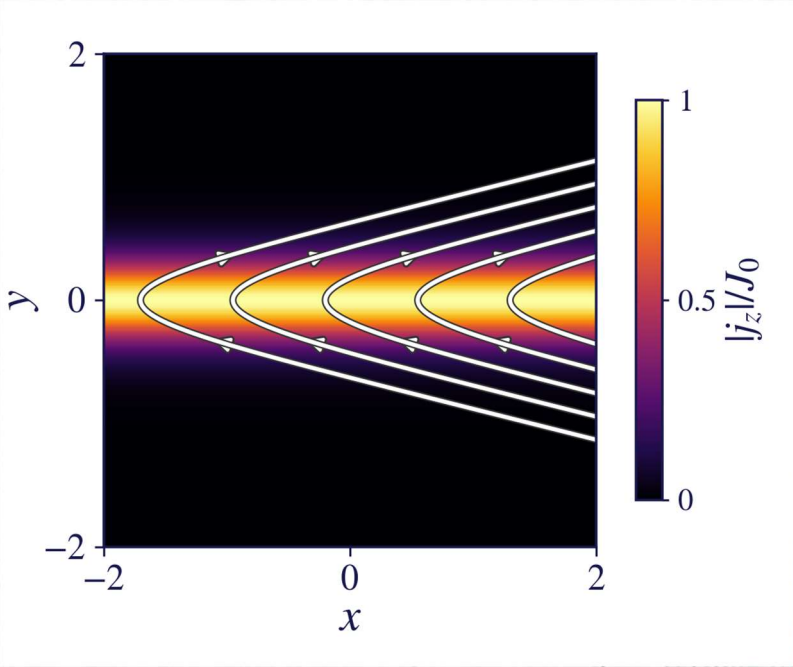
Scalar Maps with Vector Overlays

Use different visual channels for different physical quantities.

Signed magnetic component



Current magnitude + field lines



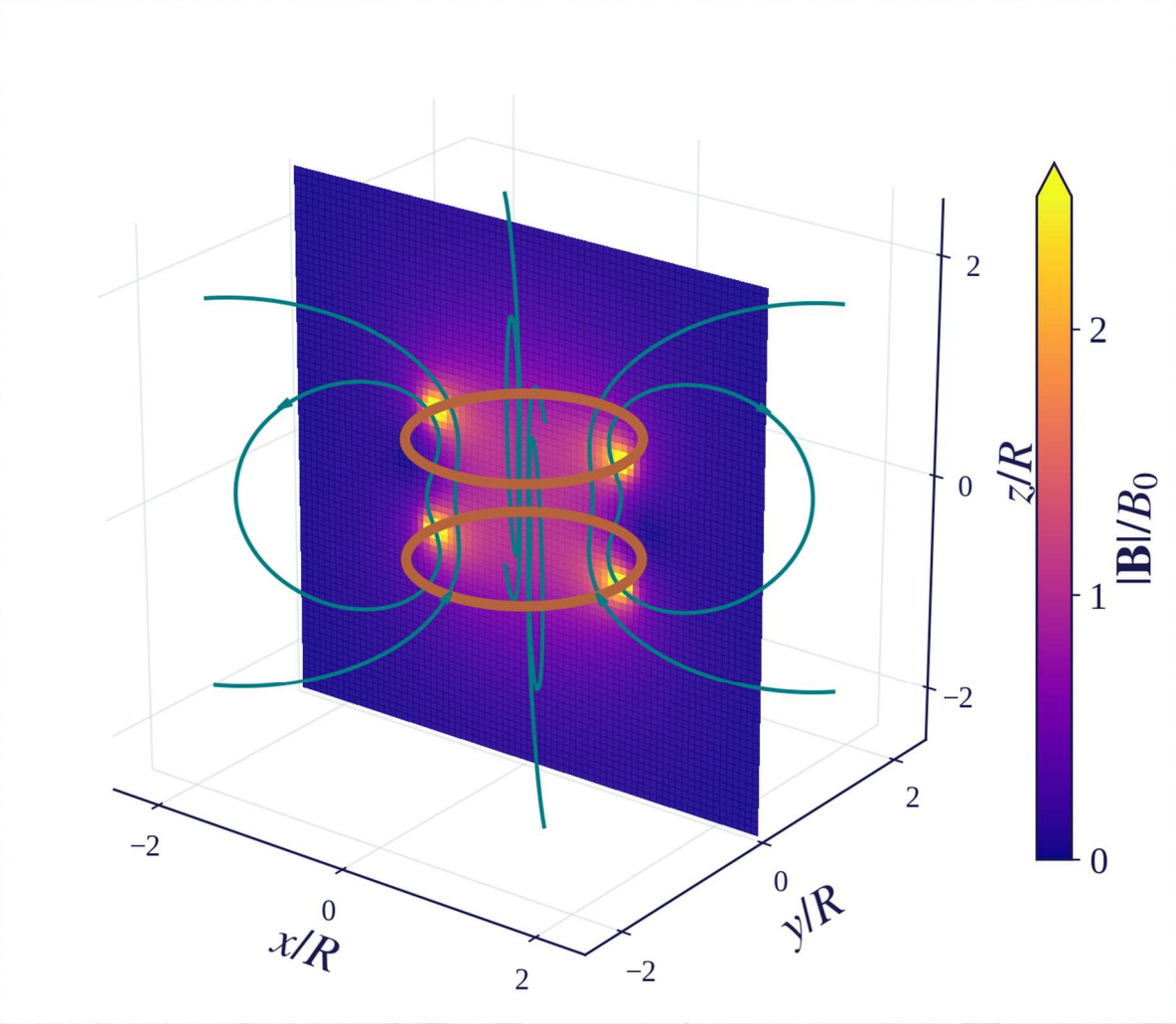
MATLAB

```
imagesc(x,y,Jabs); axis xy image; hold on
h = streamline(X,Y,Bx,By,sx,sy);
set(h,'Color','w'); colorbar
```

The current-density colourbar does not describe the field-line colours.

Biot–Savart: A Composite View

Combine conductor geometry, field lines and a scalar slice.



Copper: current loops
Teal: field lines
Slice colour: $|B|/B_0$

MATLAB

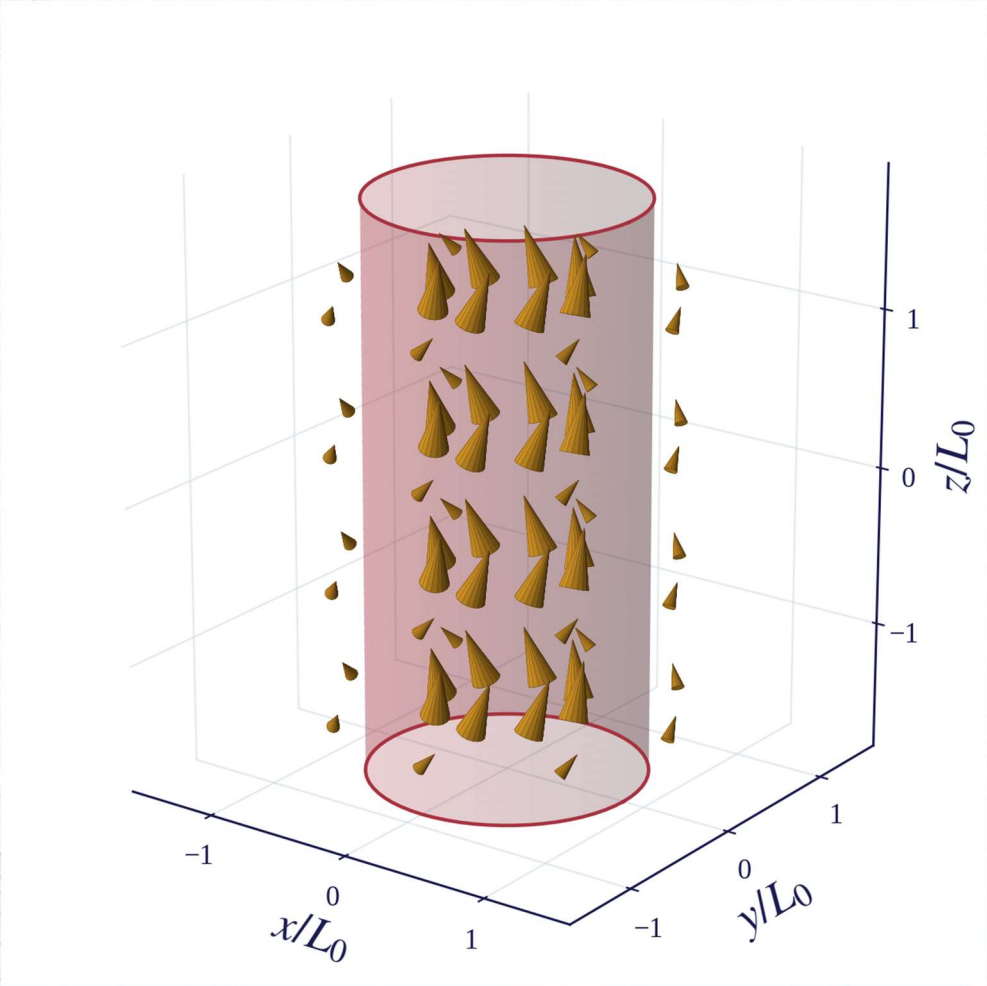
```
h = slice(X,Y,Z,Bmag,...
    [],0,[]);
set(h,'EdgeColor','none');
hold on; streamline(lines);
plot3(W(:,1),W(:,2),W(:,3));
axis equal; view(3)
```

lines: field-line vertices
W: conductor coordinates

Computed fields are inputs; the snippet assembles their graphical layers.

Cone Plots

A three-dimensional glyph makes local vector orientation visible.



Cones: direction + length
Surface: selected |B| level

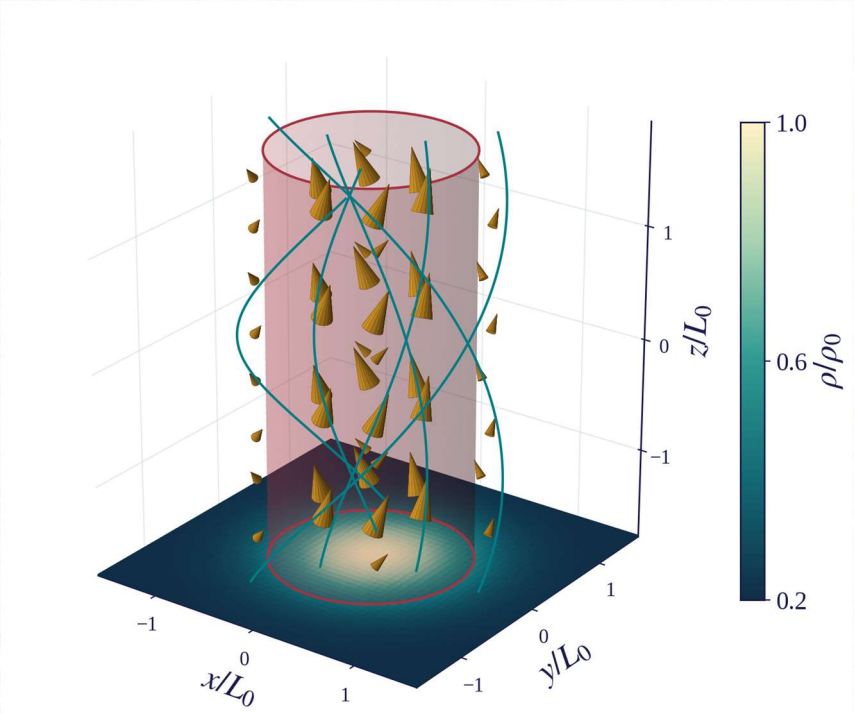
MATLAB

```
h = coneplot(X,Y,Z,...
    Bx,By,Bz,Cx,Cy,Cz,.8);
set(h,'FaceColor',gold,...
    'EdgeColor','none');
axis equal; view(3);
camlight; lighting gouraud
```

Cx, Cy, Cz choose glyph positions; the scale factor controls crowding.

Layered Three-Dimensional Views

Build the scene one layer at a time; label every encoding.



Slice → density
Surface → |B| level
Curves → field lines
Cones → local vectors

MATLAB

```
slice(X,Y,Z,Rho3,[],[],z0); hold on
p = patch(iso); set(p,'FaceColor',red,'FaceAlpha',.2);
h = streamline(lines); set(h,'Color',teal);
h = coneplot(X,Y,Z,Bx,By,Bz,Cx,Cy,Cz); h.FaceColor = gold;
```

Only the density slice uses the density colourbar; other colours identify layers.

Pressure Tensors: Two Representations

A component matrix and a geometric glyph show different aspects.

Components in a chosen basis

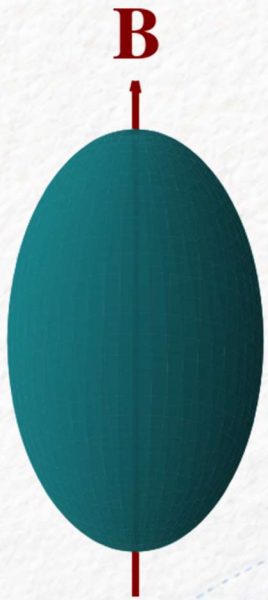
$$\begin{bmatrix} P_{xx} & P_{xy} & P_{xz} \\ P_{yx} & P_{yy} & P_{yz} \\ P_{zx} & P_{zy} & P_{zz} \end{bmatrix}$$

$P = P^T$: six independent components

MATLAB

```
P = diag([.6 .6 1.8]);
pbar = trace(P)/3;
```

Local pressure ellipsoid



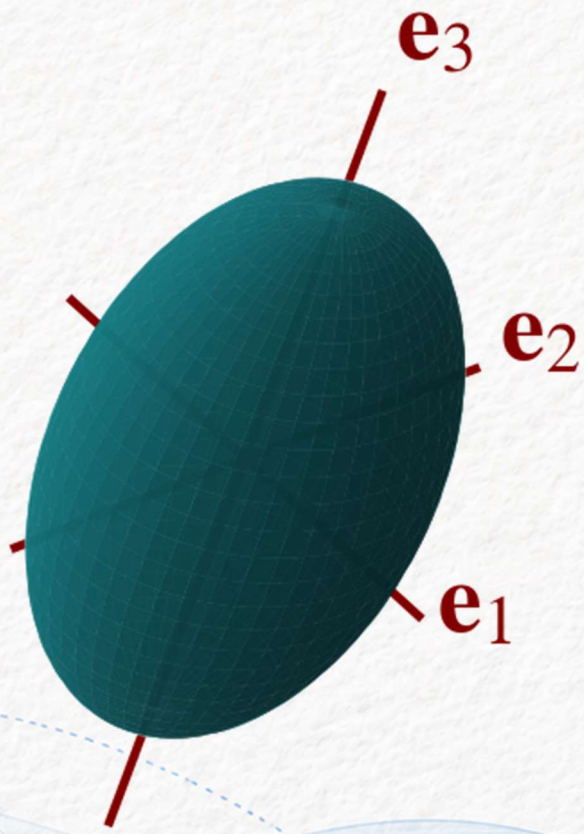
MATLAB

```
tensorGlyph(P,1,[0 0 0],1);
axis equal; view(3)
```

The ellipsoid is a graphical symbol—not a material object.

Constructing a Tensor Ellipsoid

Map a unit sphere through eigenvalue scaling and eigenvector rotation.



Eigenvectors → principal axes
Semiaxes → $\sqrt{(p_i/p_0)}$
One common glyph scale.

MATLAB

```
[V,D] = eig((P+P')/2);
r = sqrt(diag(D)/p0);
% supplied glyph helper
tensorGlyph(P,p0,[0 0 0],1);
axis equal; view(3)
```

This ellipsoid construction assumes a symmetric positive-definite tensor.

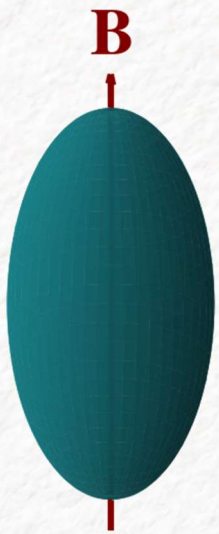
Comparing Tensor Shapes

Keep the same pressure reference and geometric scale in every panel.

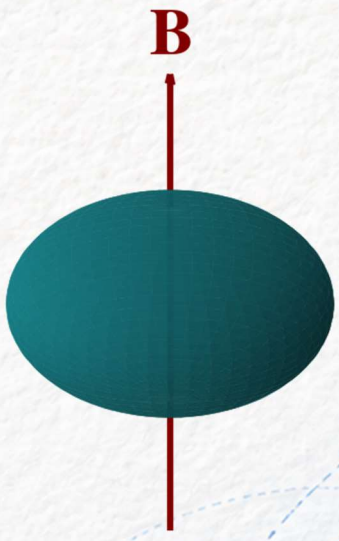
Isotropic



Parallel-dominated



Perpendicular-dominated



MATLAB

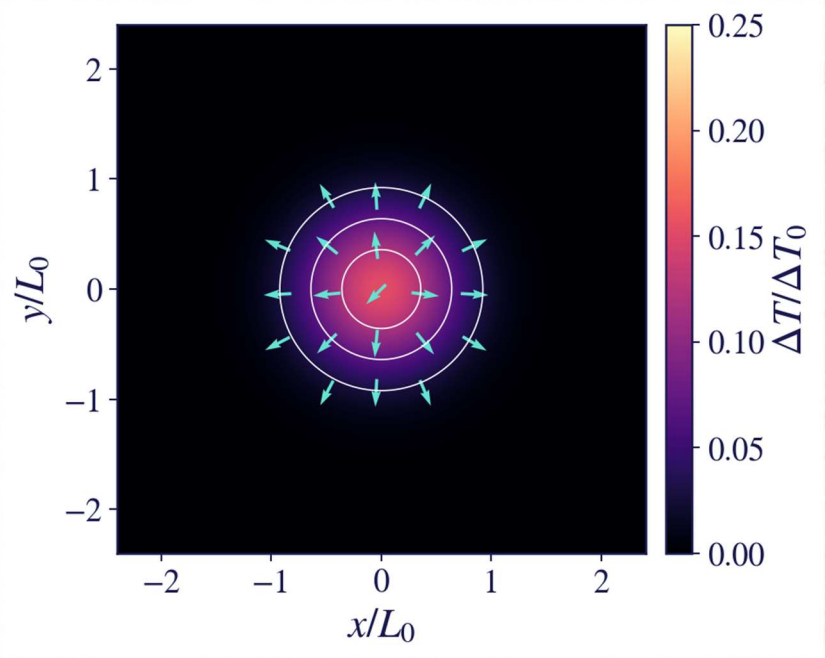
```
P = pperp*eye(3) + (ppar-pperp)*(b*b');
tensorGlyph(P,p0,centre,gain);   % same p0 and gain in every case
```

Shape encodes relative principal values; orientation encodes principal directions.

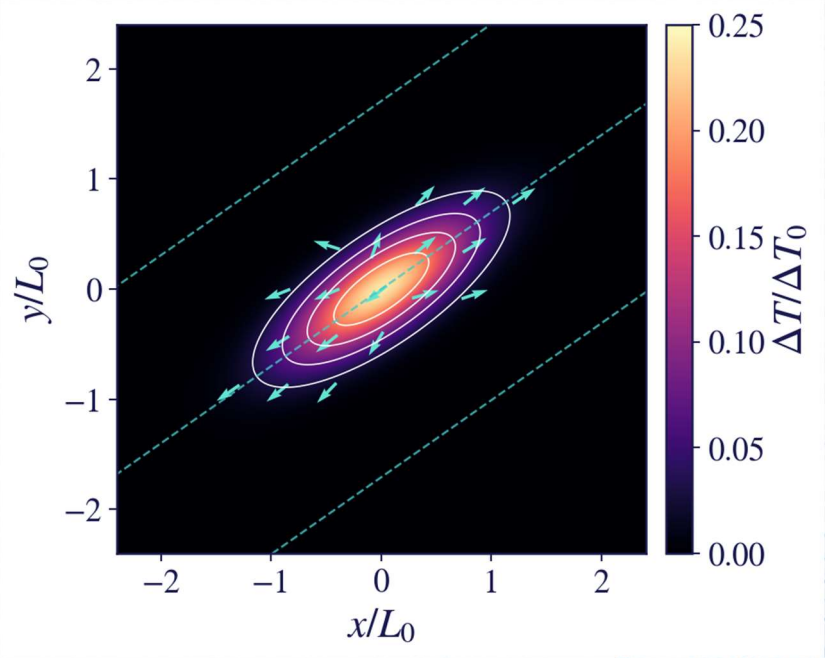
Composite Field Display

Temperature colour and contours; heat-flux arrows from a conductivity tensor.

Isotropic conductivity



Anisotropic conductivity



MATLAB

```
imagesc(x,y,T,[0 .25]); axis xy image; hold on
contour(X,Y,T,levels,'w');
quiver(Xs,Ys,Uhat,Vhat,0);    % equal-length direction arrows
```

The display combines a scalar field, a tensor-derived vector, and line geometry.

Summary: Choose the Representation

The figure is a mapping of model data—not the model itself.

Visualization methods	Array layout, scale, interpolation, camera and opacity.
Scalar fields	Maps, contours, surfaces, slices, projections and animation.
Vector fields	Arrows, field lines, LIC, cones and layered scenes.
Tensor fields	Component maps, principal axes, ellipsoids and glyph fields.

Choose the technique; state its encoding; keep the numerical arrays.

Applications

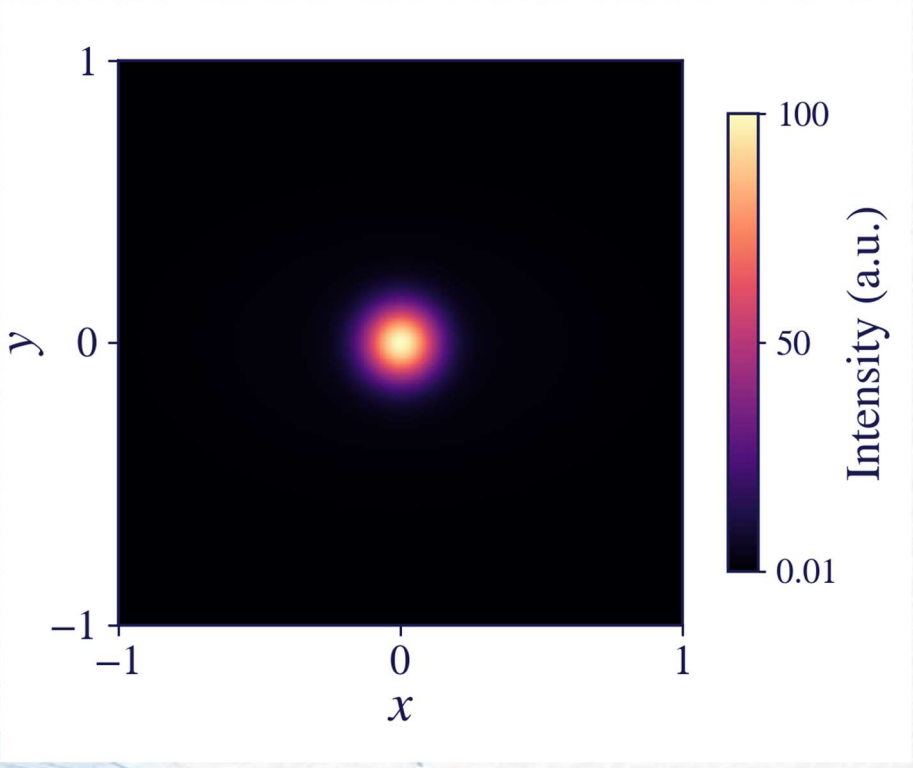
Physical diagnostics and extended exercises.

Convergence • Synthetic observations • Features • Islands

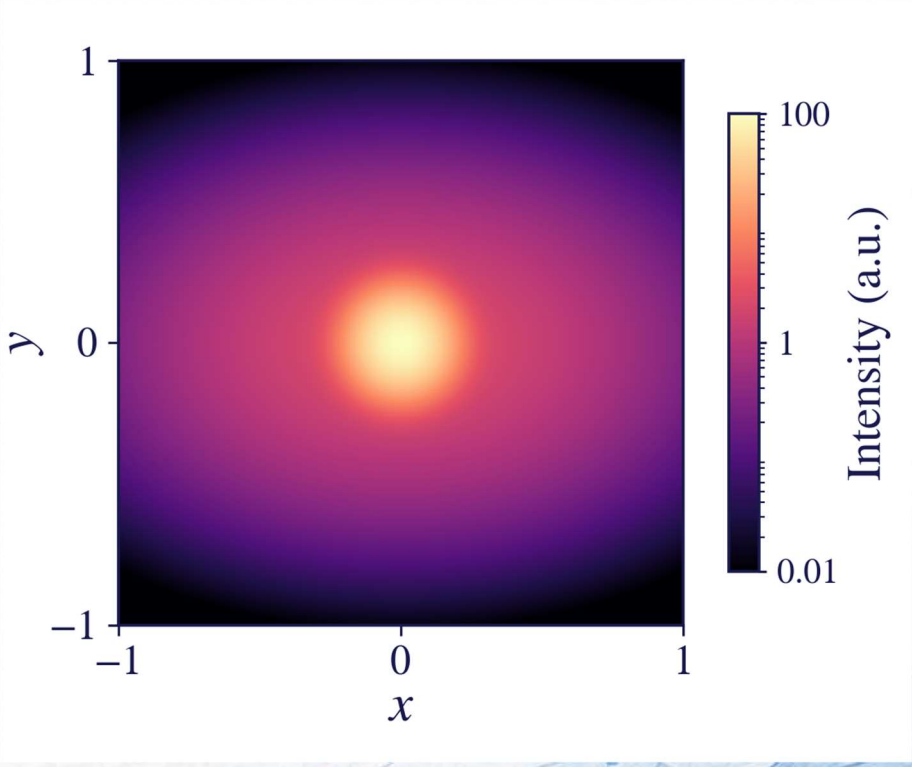
Linear and Logarithmic Scale

Same data, same palette, different mapping.

Linear



Logarithmic



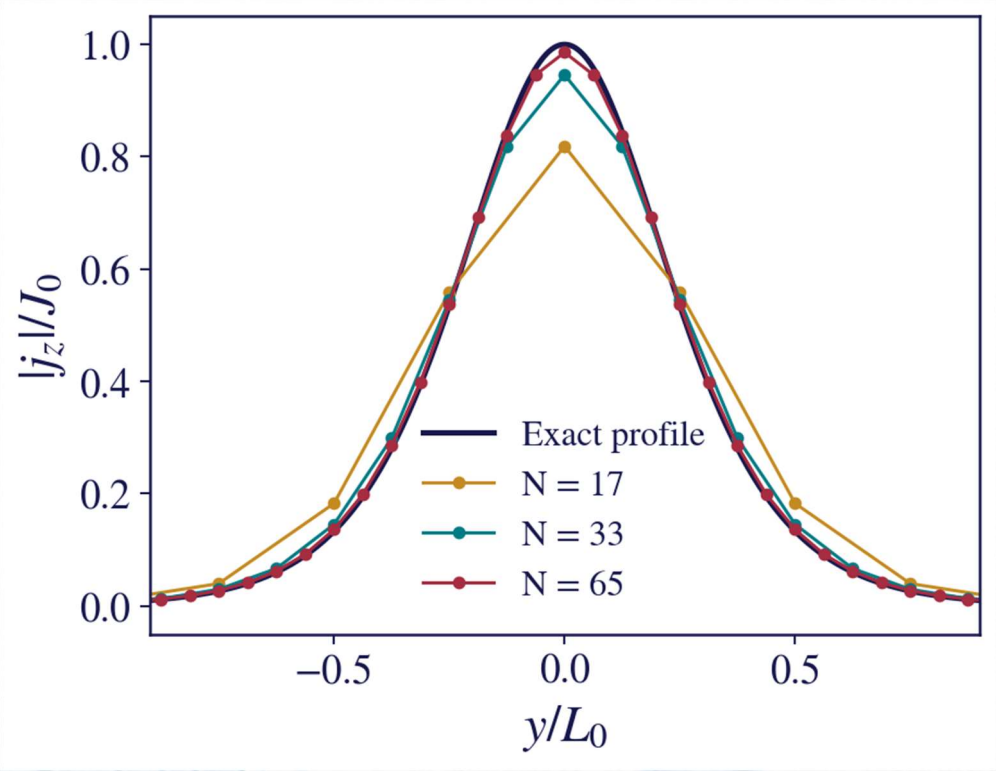

```
set(gca,'ColorScale','log')
```

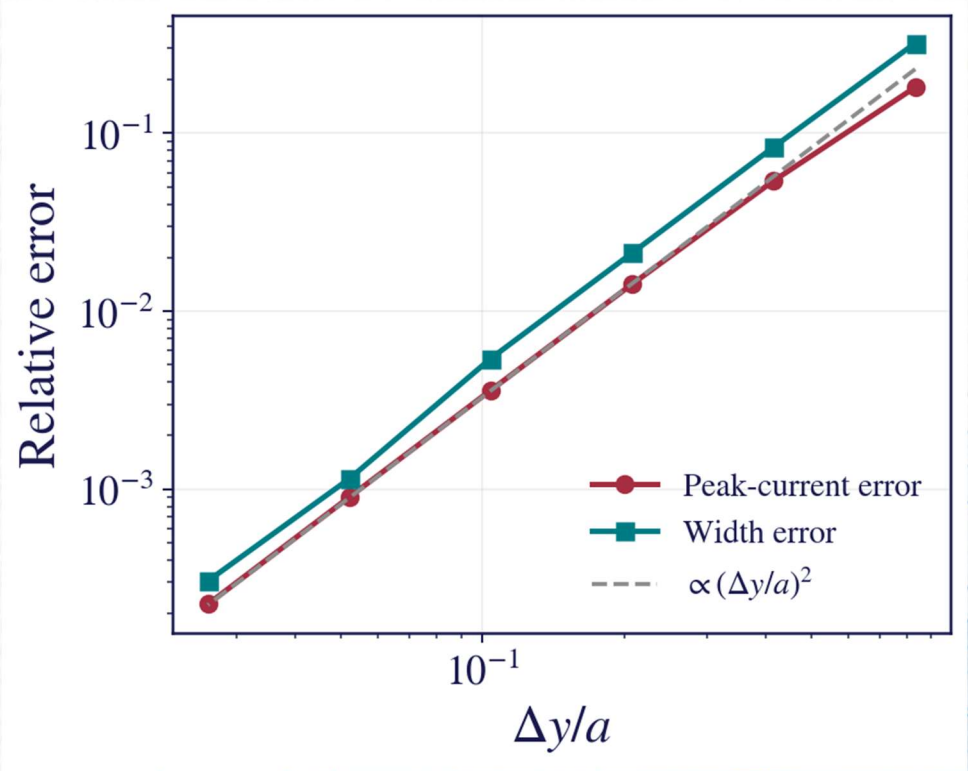
Logarithmic scaling reveals faint structure; use positive values.

Resolution and Diagnostic Convergence

Resolve the current sheet before trusting its width or peak.

$$j_z = -\frac{1}{\mu_0}\frac{\partial B_x}{\partial y}, \qquad W = 2a\,\operatorname{arcosh}\sqrt{2}$$





Halve the grid spacing → approximately four times smaller peak error.

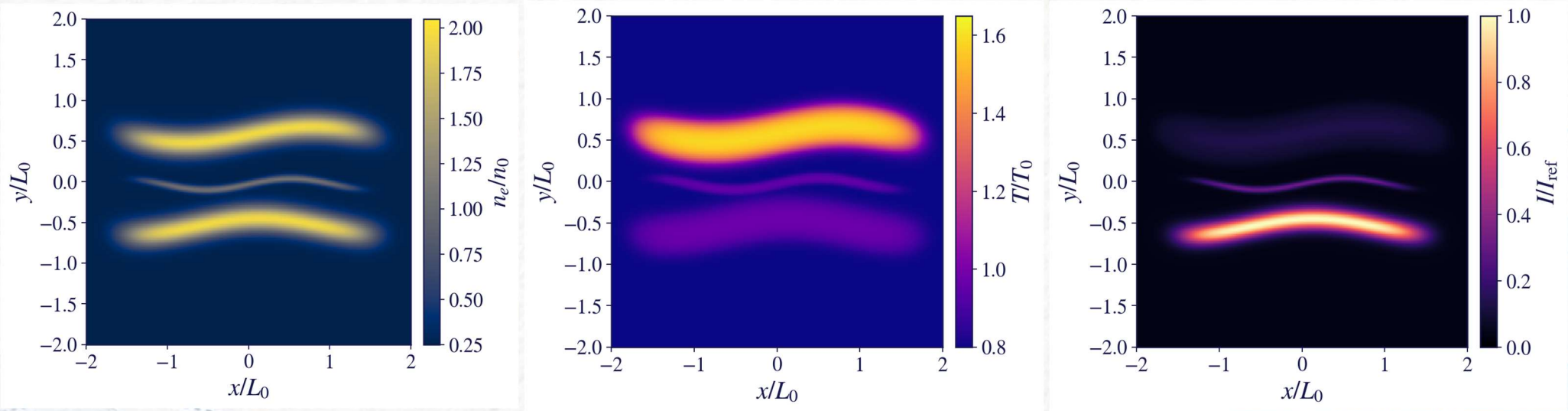
Test the measurement, not the smoothness of the image.

From Density to Synthetic Intensity

The emission model changes which structures are visible.

$$I_\lambda(x,y) = \int n_e^2 G_\lambda(T, n_e)\, \mathrm{d}z$$

Density slice | Temperature slice | Synthetic intensity



Two equally dense strands can have very different brightness.

Illustrative response G(T): not a calibrated spectral line.

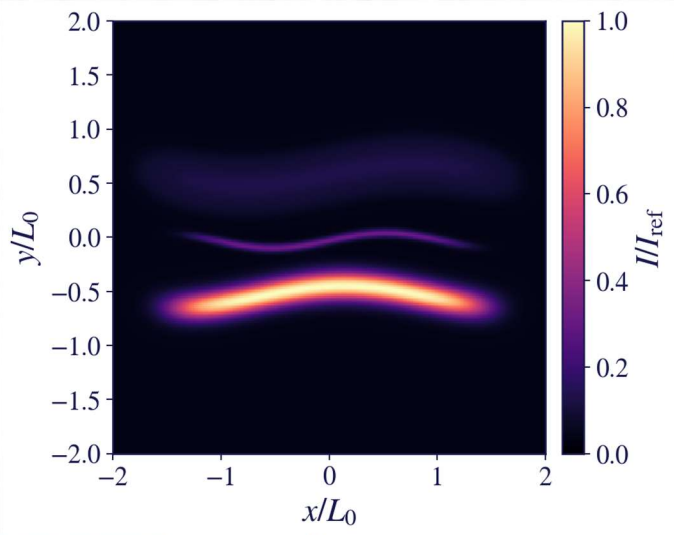
Theory: CHIANTI User Guide §6.7. The displayed response G(T) is an analytic teaching model.

From Ideal Image to Measurement

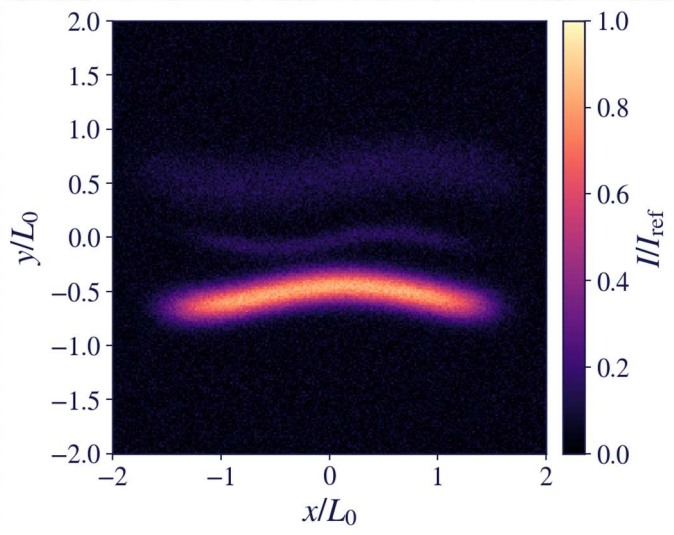
Separate the emitting structure from the instrumental response.

$$I_{\mathrm{obs}} = H * I_{\lambda} + \eta$$

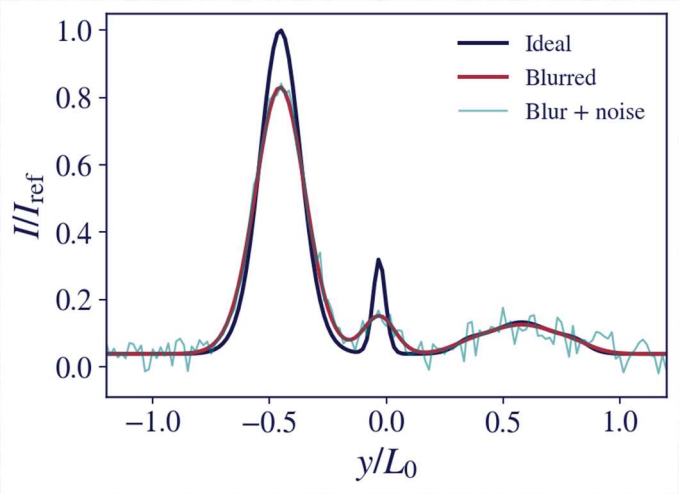
Ideal intensity



Blur + noise



Cross-strand profile



Gaussian PSF: σ = 0.06 L₀ Noise: σ = 0.03 Iref

More display pixels do not recover unresolved structure.

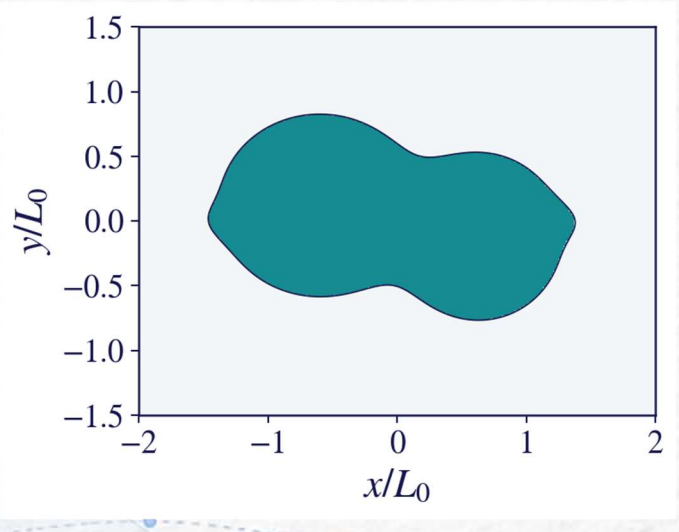
Original Gaussian-PSF and additive-noise example. No specific telescope is modelled.

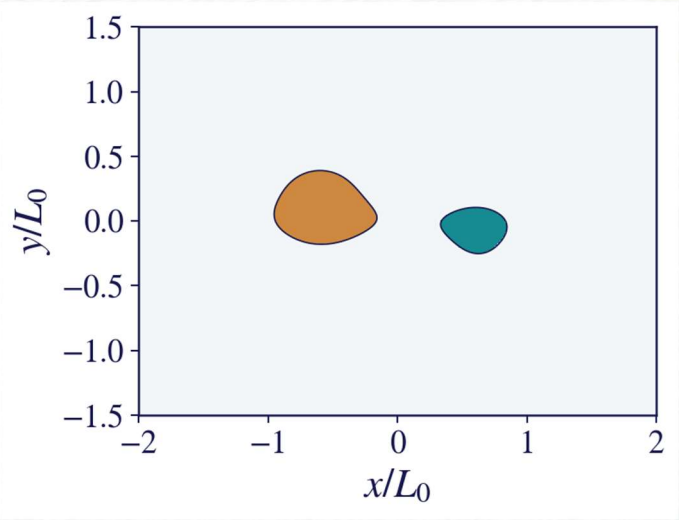
Feature Extraction and Thresholds

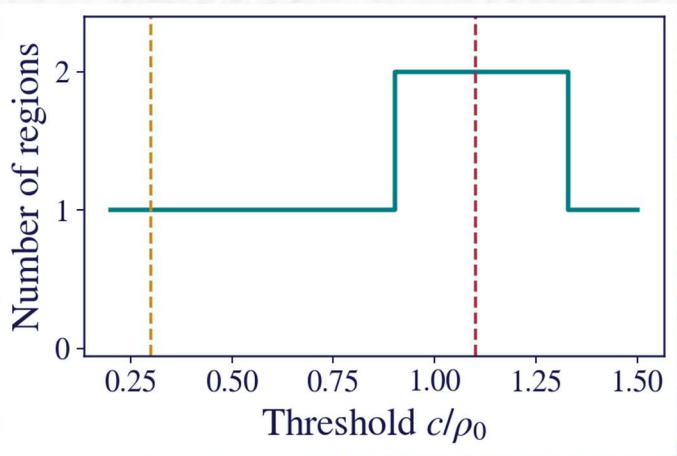
Define a structure, then test how strongly the definition matters.

$$\Omega_c = \{\mathbf{x} : \rho(\mathbf{x}) > c\}, \qquad M_c = \int_{\Omega_c} \rho \, dV$$

c = 0.30 ρ₀: one region c = 1.10 ρ₀: two regions Threshold scan







Selected mass fraction: 59.1% → 16.3% as the threshold rises.

A contour level can change the number of identified structures.

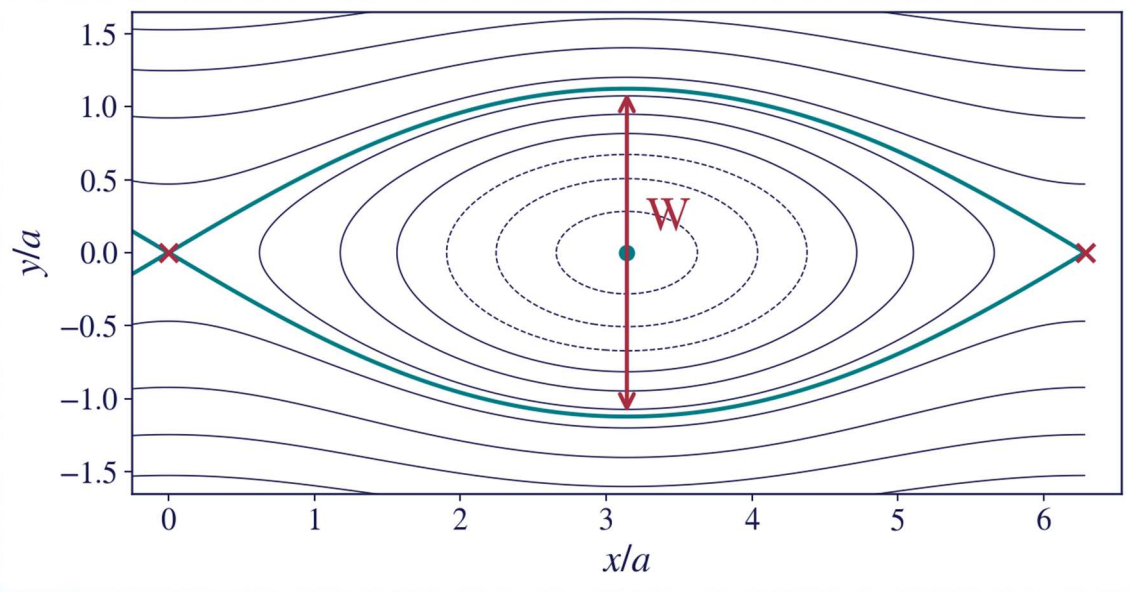
Connected components: MathWorks bwconncomp. Original two-peak density benchmark.

Measuring a Magnetic Island

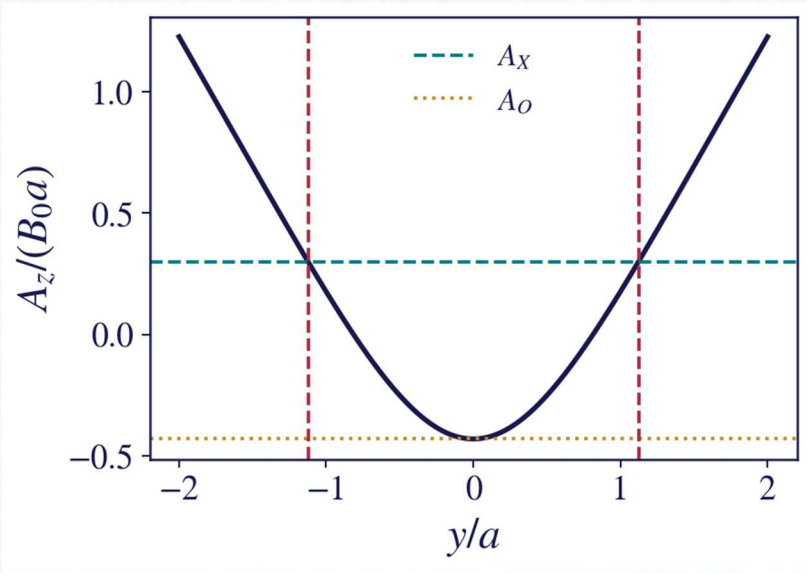
Turn magnetic geometry into a quantitative diagnostic.

$$\Delta\Psi = |A_X - A_O|, \qquad W = y_+ - y_-$$

Separatrix width



Flux profile at x = πa



ε = 0.35: W = 2.246 a ΔΨ = 0.731 B₀a

Measure the numerical field; test sensitivity to the grid.

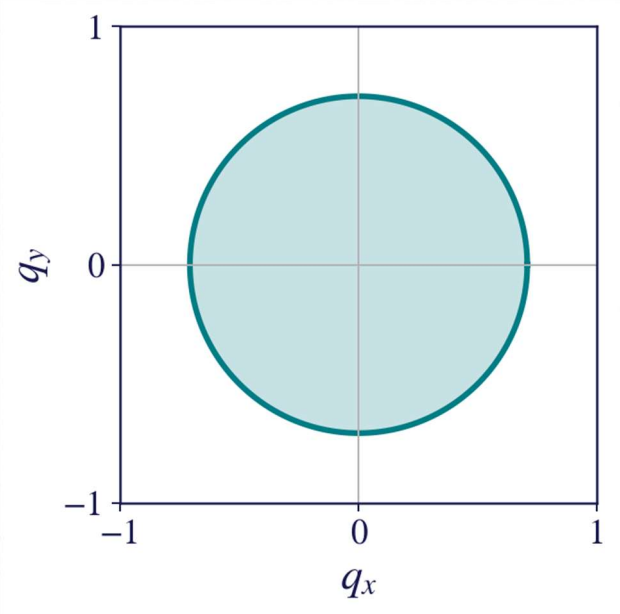
Original analytic benchmark. ΔΨ is flux per unit length along the invariant direction.

Anisotropy Is Not Agyrotropy

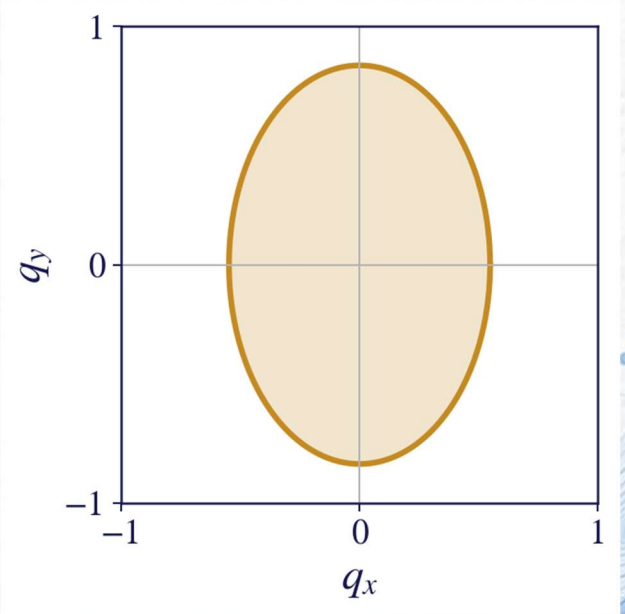
The same pressure ratio can hide different perpendicular structure.

$$\mathbf{P}_1/p_0 = \mathrm{diag}(0.5, 0.5, 2) \quad \mathbf{P}_2/p_0 = \mathrm{diag}(0.3, 0.7, 2)$$

Gyrotropic



Agyrotropic



Perpendicular pressure sections; B is normal to the image plane.

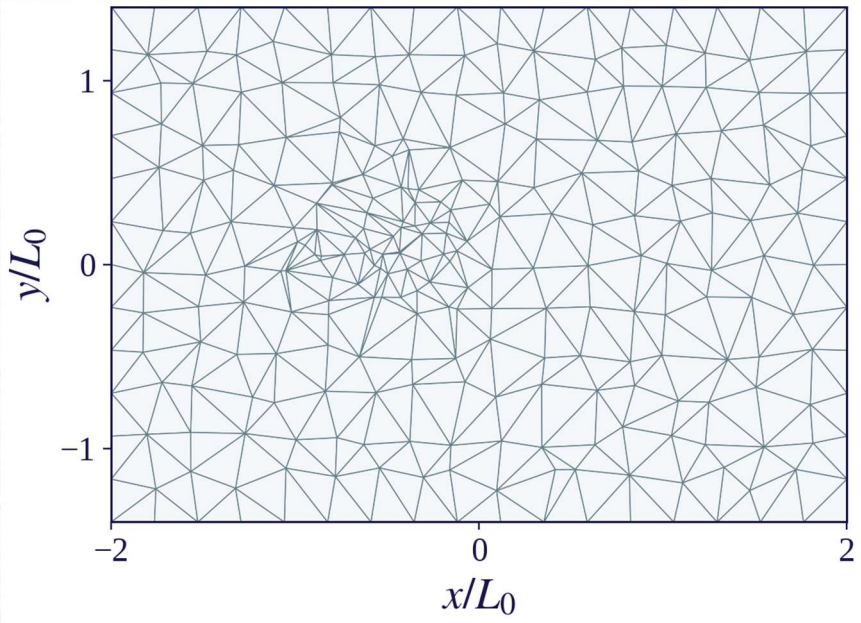
$$p_{\parallel}/\overline{p}_{\perp} = 4 \quad \text{in both cases}$$

Distinction and limitations: Che et al. (2018); Swisdak (2016). Original comparison tensors.

Unstructured Grids and Cell Data

Plot directly on the numerical mesh—not on an invented image grid.

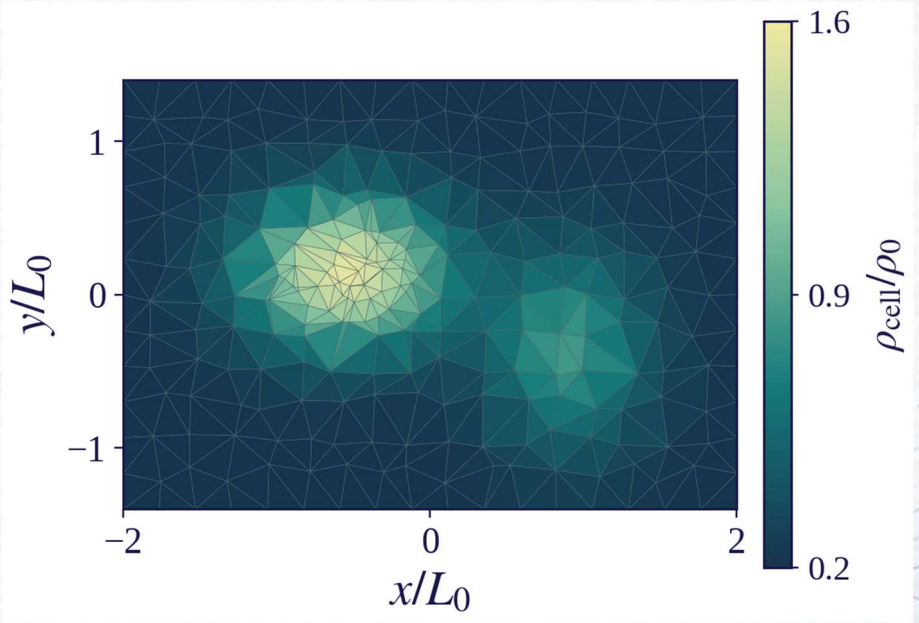
Mesh connectivity



MATLAB

```
triplot(tri,xy(:,1),xy(:,2));
axis equal tight
% tri: one row per triangle
```

One density value per cell



MATLAB

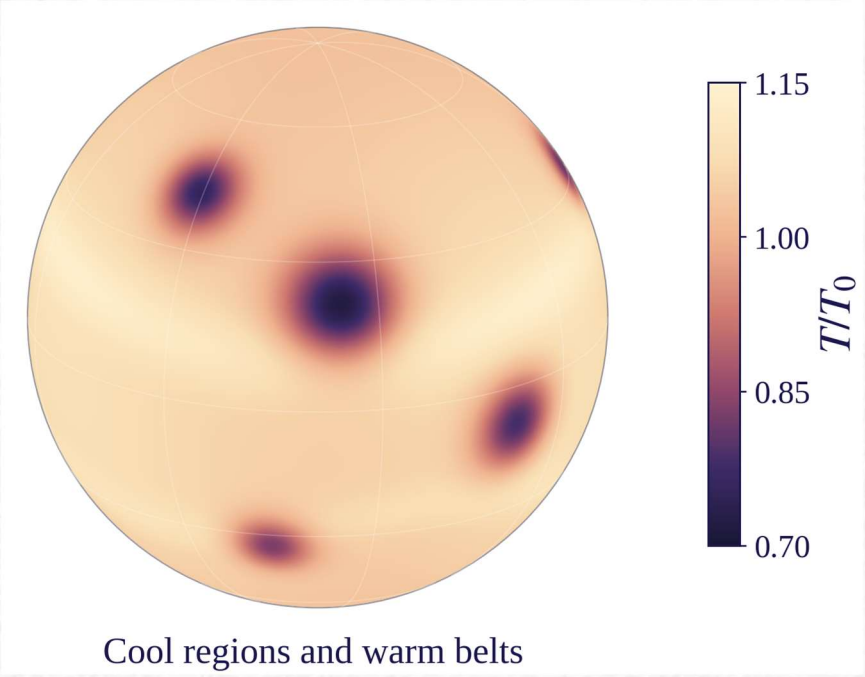
```
p = patch('Faces',tri,'Vertices',xy);
p.FaceVertexCData = rhoCell(:);
p.FaceColor = 'flat';
p.EdgeColor = 'none'; colorbar
```

Cell colours need cell values; retain the original mesh connectivity.

Geometry and Colour

The coordinates define the surface; colour displays an independent field.

Temperature: cool spots and belts

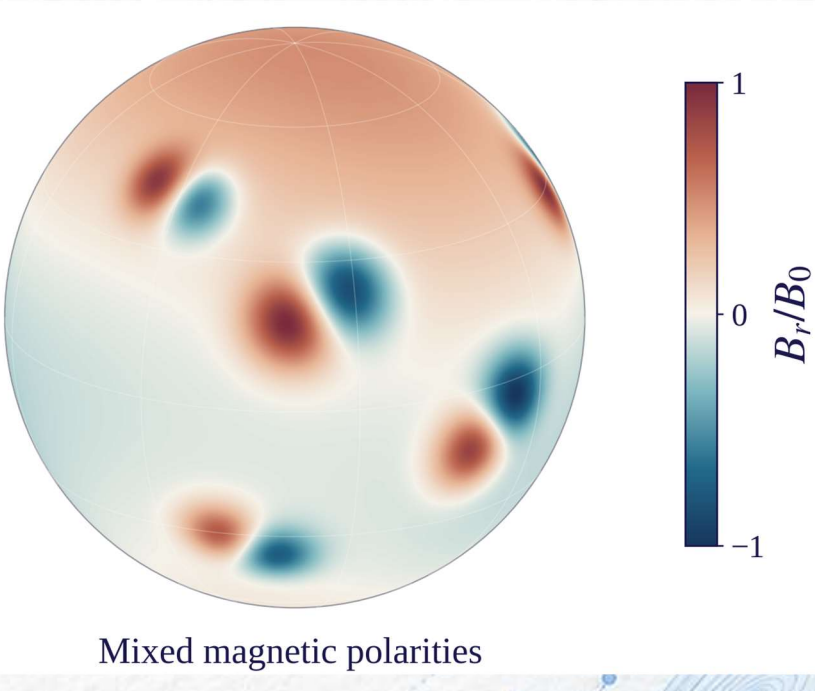


Cool regions and warm belts

MATLAB

```
surf(X,Y,Z,T,'EdgeColor','none');
colormap(cmTemp); clim([.70 1.15]);
axis equal; view(-58,19); colorbar
```

Radial field: mixed polarities



Mixed magnetic polarities

MATLAB

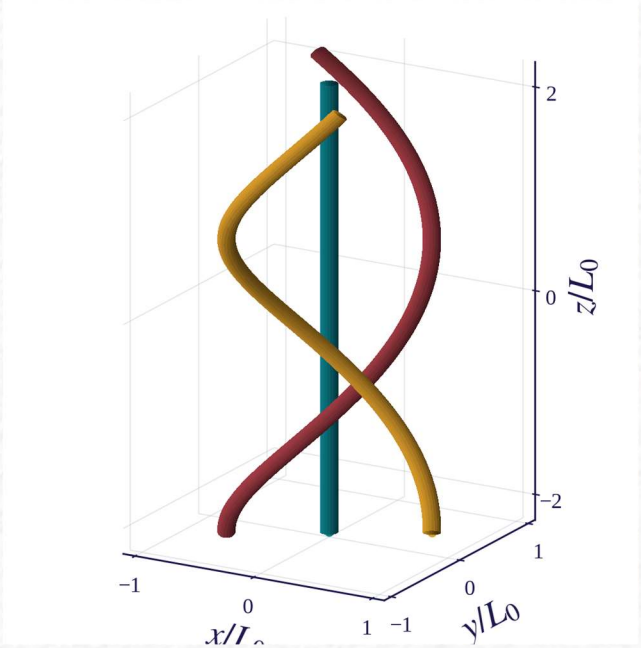
```
surf(X,Y,Z,Br,'EdgeColor','none');
colormap(cmField); clim([-1 1]);
axis equal; view(-58,19); colorbar
```

Identical geometry; colour reveals different surface structures.

Streamtubes and Stream Ribbons

The same velocity field, seeds and camera—with two curve representations.

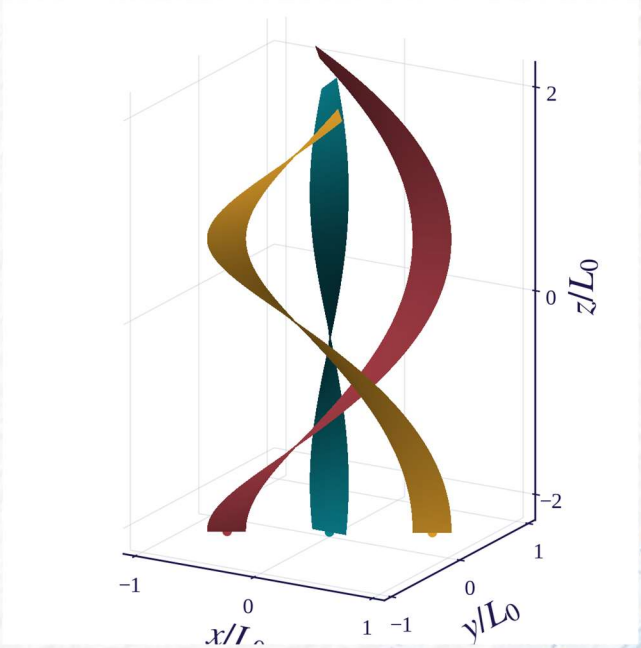
Tubes: fixed display diameter



MATLAB

```
verts = stream3(X,Y,Z,U,V,W,...
    sx,sy,sz);
streamtube(verts,0.13);
axis equal; view(3)
```

Ribbons: curl-based rotation



MATLAB

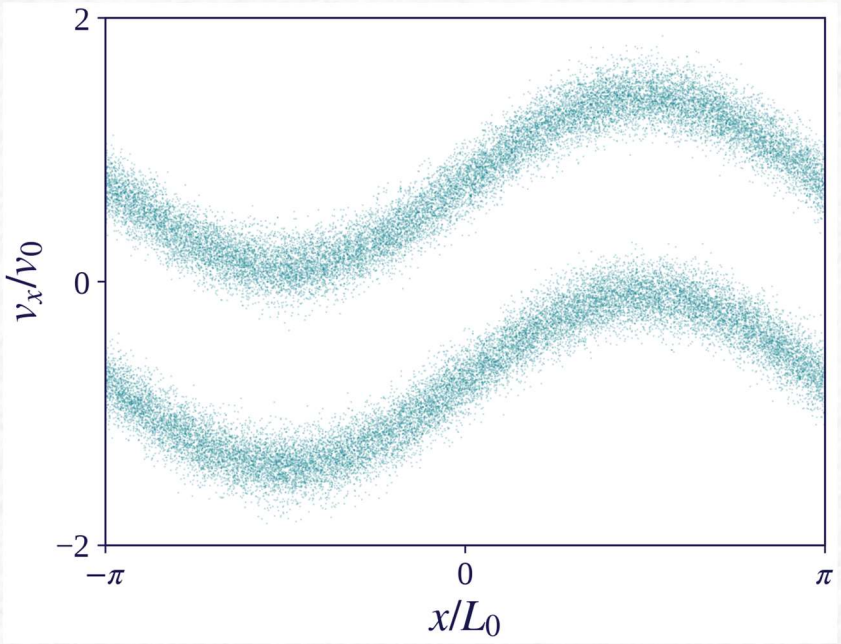
```
streamribbon(X,Y,Z,U,V,W,...
    sx,sy,sz,0.28);
axis equal; view(3);
camlight; lighting gouraud
```

Tube width is fixed here; ribbon twist is not a speed colour scale.

Particles and Phase Space

40 000 equal-weight particles: individual samples or counts per bin.

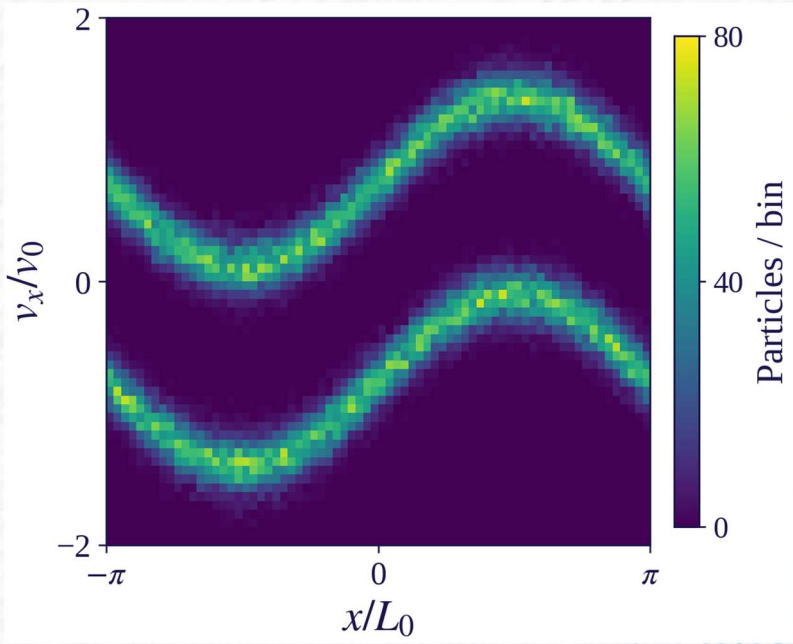
Scatter: individual particles



MATLAB

```
scatter(xp,vxp,3,'.');
xlabel('x/L_0'); ylabel('v_x/v_0');
% 40 000 equal-weight particles
```

Histogram: particles per bin



MATLAB

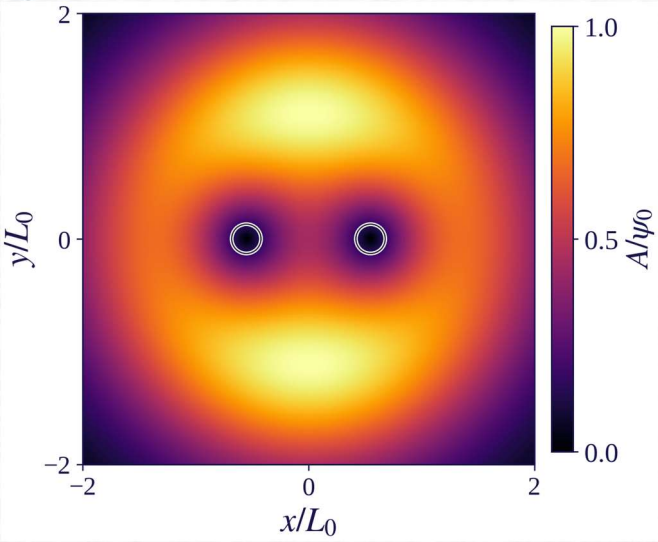
```
histogram2(xp,vxp,xEdges,vEdges,...
    'DisplayStyle','tile',...
    'ShowEmptyBins','on');
colorbar
```

Same samples: 72 × 60 bins. Counts are not yet a normalized distribution.

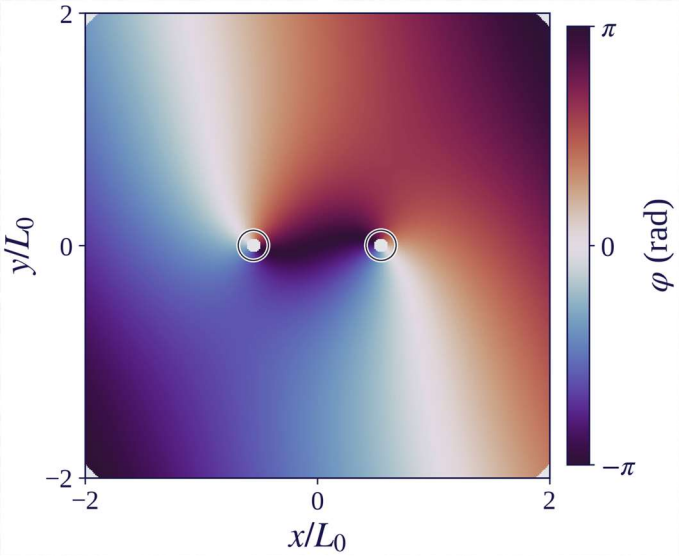
Complex Fields: Reading the Phase

$\psi = A \exp(i\varphi)$: amplitude gives strength; phase gives position in the cycle.

Amplitude $A = |\psi|$

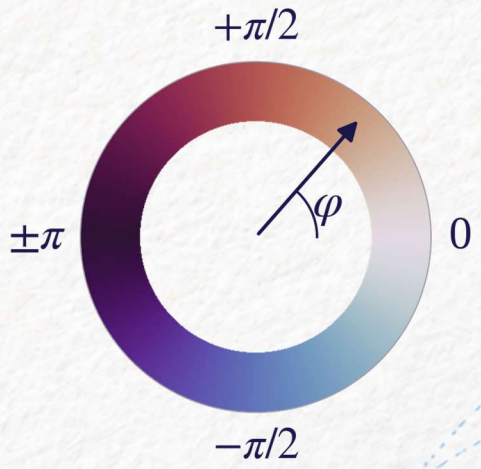


Wrapped phase $\varphi = \arg \psi$



Circles mark $A = 0$: the phase is undefined there.

Read phase as an angle



$-\pi$ and $+\pi$ are identical.

Grey: $A < 0.08\ \psi_0$.

MATLAB

```
A = abs(psi);
imagesc(x,y,A,[0 1]);
axis xy image
colormap(cmAmp); colorbar
```

MATLAB

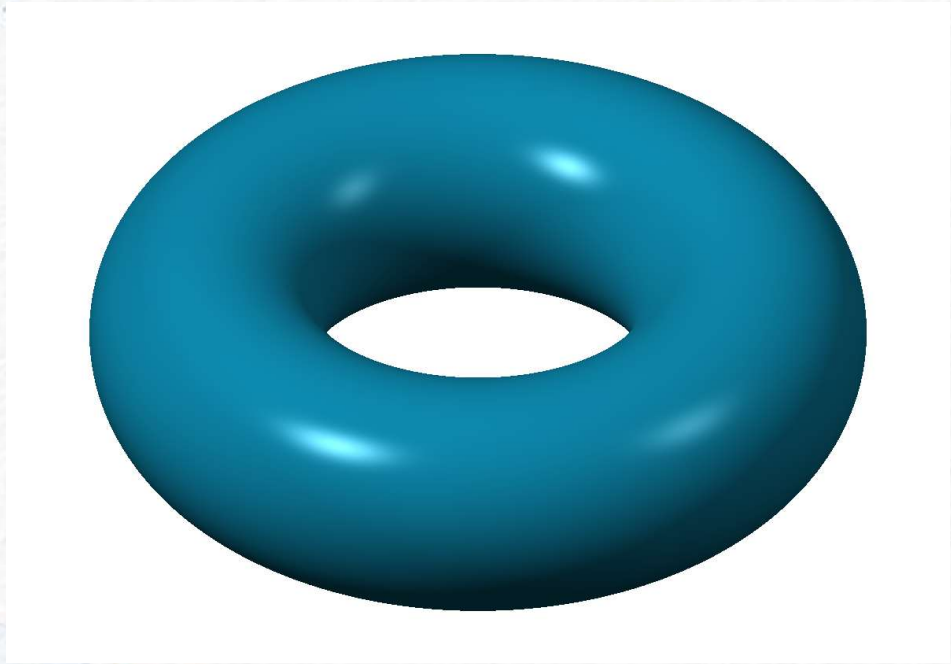
```
phi = angle(psi);
h = imagesc(x,y,phi); h.AlphaData = abs(psi)>0.08;
set(gca,'Color',[.89 .90 .91]);
axis xy image; clim([-pi pi]); colormap(cmPhase)
```

A 2π wrap in the phase does not imply a jump in the complex field.

Transparency Reveals Interior Surfaces

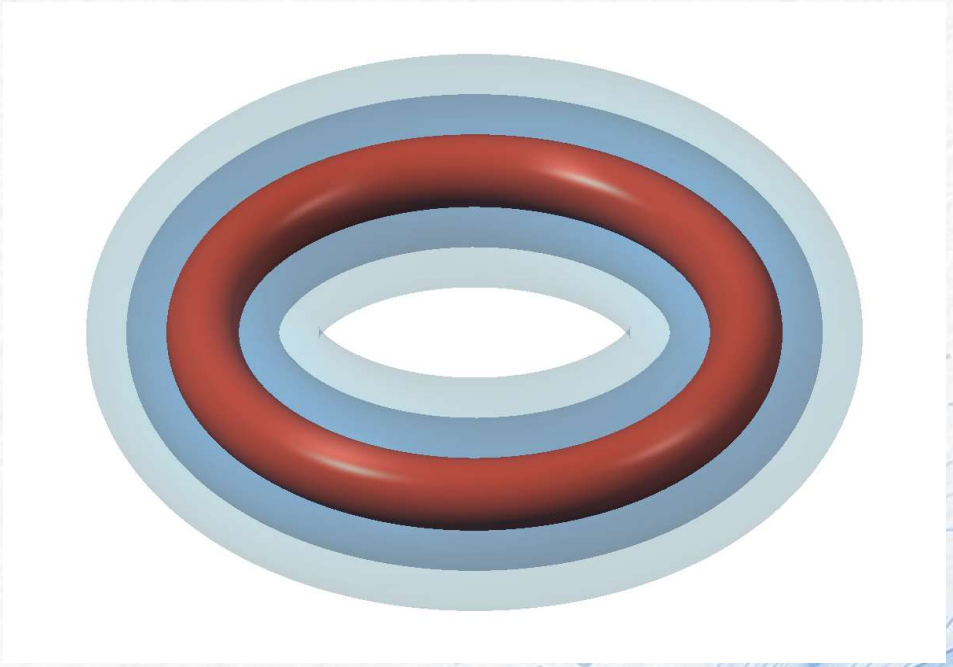
Same three closed toroidal surfaces, same camera—only the outer opacity changes.

Opaque outer torus



```
MATLAB
% h(1:3): outer, middle, core
set(h(1),'FaceAlpha',1);
camlight; lighting gouraud
```

Transparent shell: inner core is visible



```
MATLAB
set(h(1),'FaceAlpha',.13);
set(h(2),'FaceAlpha',.22);
set(h(3),'FaceAlpha',1);
```

Colours identify surfaces; transparency is a display setting, not physical optical depth.

Exercises in Scientific Visualization

Construct every numerical array from the stated analytic definitions. No external data files are required. All quantities are dimensionless unless otherwise specified.

For each problem, submit the construction and rendering code, figures with declared scales, and concise answers to the questions. Keep the distinction between model data and display settings explicit.

Seven problems · MATLAB or an equivalent numerical language

Problem 1. Representations of an Analytic Field

On $\Omega = [-2\pi, 2\pi] \times [-\pi, \pi]$, define the following field with $\varepsilon = 0.35$.

$$D = \cosh y + \varepsilon \cos x, \quad A_z = \ln D, \quad \mathbf{B} = \left(\frac{\sinh y}{D}, \frac{\varepsilon \sin x}{D}, 0 \right)$$

$$j_z = -\frac{1 - \varepsilon^2}{D^2}, \qquad \varepsilon = 0.35, \qquad B_0 = a = \mu_0 = 1$$

Sample x at 161 points and y at 81 points. All quantities are dimensionless.

(a) Construct maps of A_z and |j_z|, labelled contours, and a surface plot. State what height, colour and contour level represent.

(b) Display the same B using sparse arrows and seeded field lines. Retain the same coordinates and aspect ratio.

(c) Combine a current-density map with flux contours and field-line curves. Explain every graphical layer.

Question. Which information is omitted by each representation?

Problem 2. Magnetic Reversal and Line Profiles

Let $(x,y) \in [-2,2]^2$ and use a uniform 301×301 grid with $B_0 = \mu_0 = 1$.

$$B_x = \tanh(y/a), \quad B_y = 0.25, \quad j_z = -a^{-1}\operatorname{sech}^2(y/a), \quad a = 0.30$$

Keep the same reference current Jref = 1/0.30 in every comparison.

(a) Plot B_x with symmetric colour limits and overlay magnetic-field lines on a map of |j_z|.

(b) Extract the profile at x = 0. Estimate its full width at half maximum by interpolating the two crossings.

(c) Repeat for a = 0.15 without renormalizing either run by its own maximum. Compare peak current and width.

Question. How can independent normalization conceal a physical change in amplitude?

Problem 3. Flux Geometry and Grid Refinement

Use the analytic island field of Problem 1 with ε = 0.35.

$$\Delta\Psi = |A_X - A_O|, \qquad A_X = \ln(1+\varepsilon), \qquad A_O = \ln(1-\varepsilon)$$

Construct new arrays at Nx = 81, 161, 321 and 641; set Ny = (Nx + 1)/2.

(a) Locate the O-point at (π,0), the X-point at (0,0), and the contour A_z = A_X. Distinguish flux contours from density contours.

(b) Measure the transverse island width on x = π and the flux difference ΔΨ. Compare all four sampling grids.

(c) Calculate j_z from numerical derivatives of B and compare with the analytic expression. Plot the relative errors.

Question. Why does this static field not determine a reconnection rate?

Problem 4. Multiscale Sampling and Interpolation

On $[-2,2]^2$, define the scalar field by the expressions below.

$$g_1 = \exp\left[-\left(\frac{y - 0.38\sin(2.2x) - 0.15\cos(3.5x)}{0.16}\right)^2\right]$$

$$g_2 = \exp\left[-\left(\frac{y + 0.85 + 0.18\sin(3.1x)}{0.23}\right)^2\right], \quad q = \sqrt{\left(\frac{x+0.9}{0.72}\right)^2 + \left(\frac{y-0.9}{0.58}\right)^2}$$

$$g_3 = e^{-[(q-1)/0.23]^2}, \quad g_4 = e^{-[(x-1.05)/0.38]^2 - [(y-0.95)/0.43]^2}, \quad g_5 = [1 + \sin(4x + 0.9y)]e^{-0.6(x^2+y^2)}$$

$$f(x,y) = 0.08 + 0.65g_1 + 0.47g_2 + 0.88g_3 + 0.72g_4 + 0.20g_5$$

Use 29 × 29 samples; evaluate the same formula separately on a 401 × 401 reference grid.

(a) Compare native samples, linear interpolation and cubic interpolation using identical axes, limits and contour levels.
(b) Normalize all arrays by the coarse-grid maximum. Plot errors against the analytic reference; report RMSE and overshoot.
(c) Identify features whose appearance depends strongly on interpolation rather than on resolved model samples.

Question. Does a smooth image demonstrate adequate spatial resolution?

Problem 5. Nested Surfaces and Visibility

For $0 \le u,v \le 2\pi$, construct three closed tori with a common major radius.

$$x = (R + r\cos v)\cos u, \quad y = (R + r\cos v)\sin u, \quad z = r\sin v$$

Use R = 2.1 and r ∈ {0.90, 0.59, 0.28}; sample 241 × 121 parameter points.

(a) Construct the surfaces analytically. Compare opaque and transparent views using fixed categorical colours.

(b) Vary outer opacity, light direction and camera position separately. Record the settings and identify occluded surfaces.

(c) Export an eight-second turntable at 24 frames per second. Preserve the geometry, opacity, colour mapping and frame size.

Question. How does display opacity differ from physical optical depth?

Problem 6. Vector Encodings and Field Reversal

Construct the analytic island field on the grid specified in Problem 1.

$$D = \cosh y + \varepsilon \cos x, \quad A_z = \ln D, \quad \mathbf{B} = \left(\frac{\sinh y}{D}, \frac{\varepsilon \sin x}{D}, 0 \right)$$

Use equal physical grid spacing. For LIC comparisons, retain the same input noise realization.

(a) Compare magnitude-scaled arrows, seeded field lines and LIC for exactly the same B. Declare all normalization choices.

(b) Vary arrow spacing, seed positions and LIC kernel length. Reverse B while holding the graphical settings fixed.

(c) Check tangency of the traced curves to B and compare them with A_z contours. State which encodings preserve vector sign.

Question. Why is a symmetric LIC texture insensitive to B → −B?

Problem 7. Tensor Glyphs and Rotation

At 81 points on a uniform 9 × 9 grid in $[-2,2]^2$, define the tensors below.

$$\mathbf{P} = \mathbf{R}\,\mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)\mathbf{R}^T, \quad \mathbf{R} = R_z(\theta)R_y(\phi)$$

$$\theta = 0.5x + 0.25y, \quad \phi = 0.18\sin x, \quad s = 1 + 0.25\cos x\cos y$$

$$(\lambda_1, \lambda_2, \lambda_3) = s(0.4 + 0.1\sin x,\ 1,\ 1.6 - 0.1\sin x)$$

Rz and Ry are standard active rotation matrices. Use $p_0 = 1$ and one glyph scale.

(a) Construct ellipsoids with semiaxes $\sqrt{(\lambda_i/p_0)}$. Verify symmetry and positive eigenvalues at every grid point.

(b) Compare P with its shape-normalized form P/(tr P/3). In the normalized view, encode the removed mean pressure by colour.

(c) Animate $P(\alpha) = Rz(\alpha)\ P\ Rz(\alpha)^T$ through one full turn. Fix the camera and check preservation of trace and eigenvalues.

Question. Which changes are orientation changes, and which change intrinsic tensor shape?