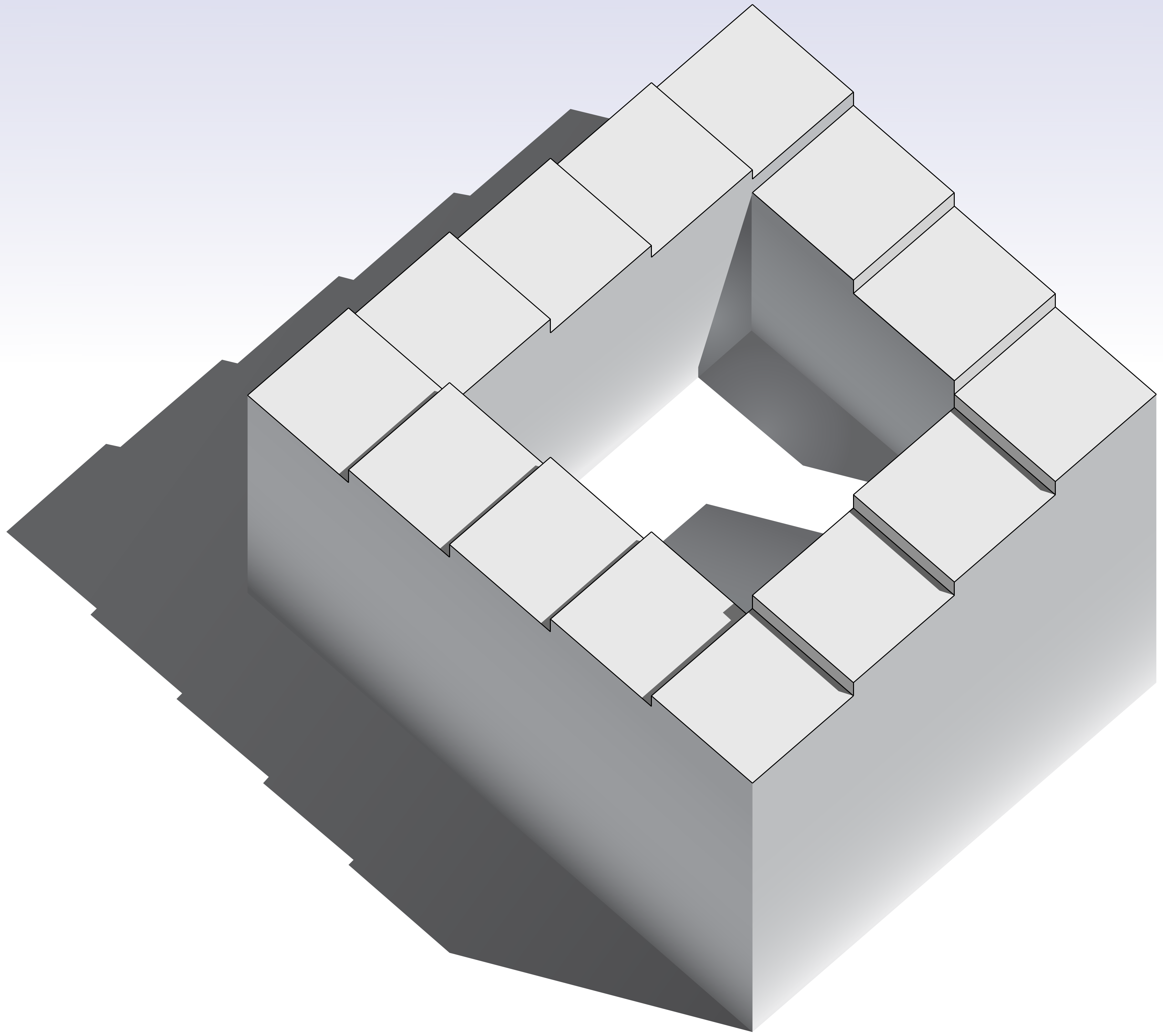
SIGGRAPH2013



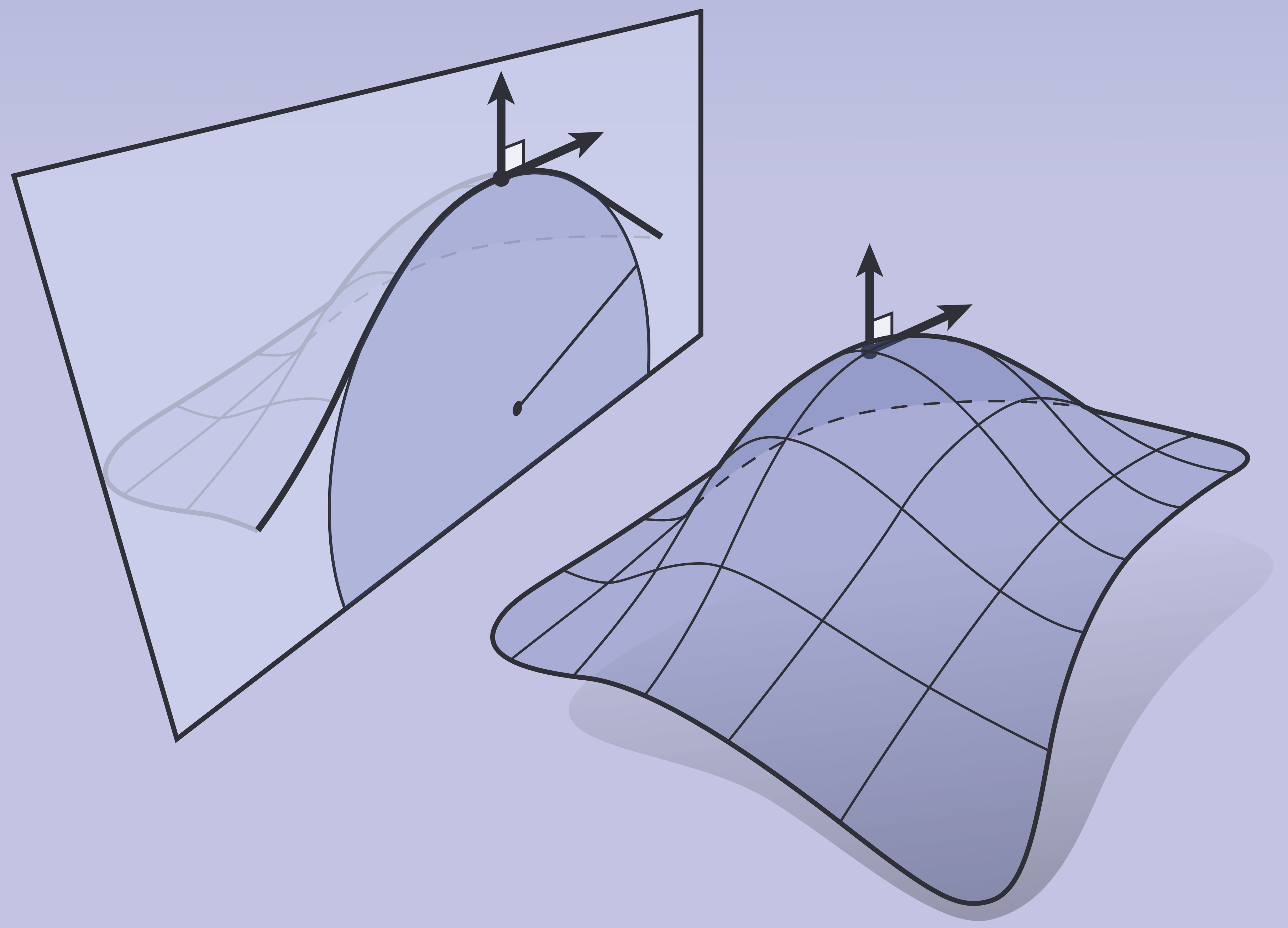
DIGITAL GEOMETRY PROCESSING WITH DISCRETE EXTERIOR CALCULUS

Keenan Crane • Fernando de Goes • Mathieu Desbrun • Peter Schröder

SIGGRAPH 2013

Part II:

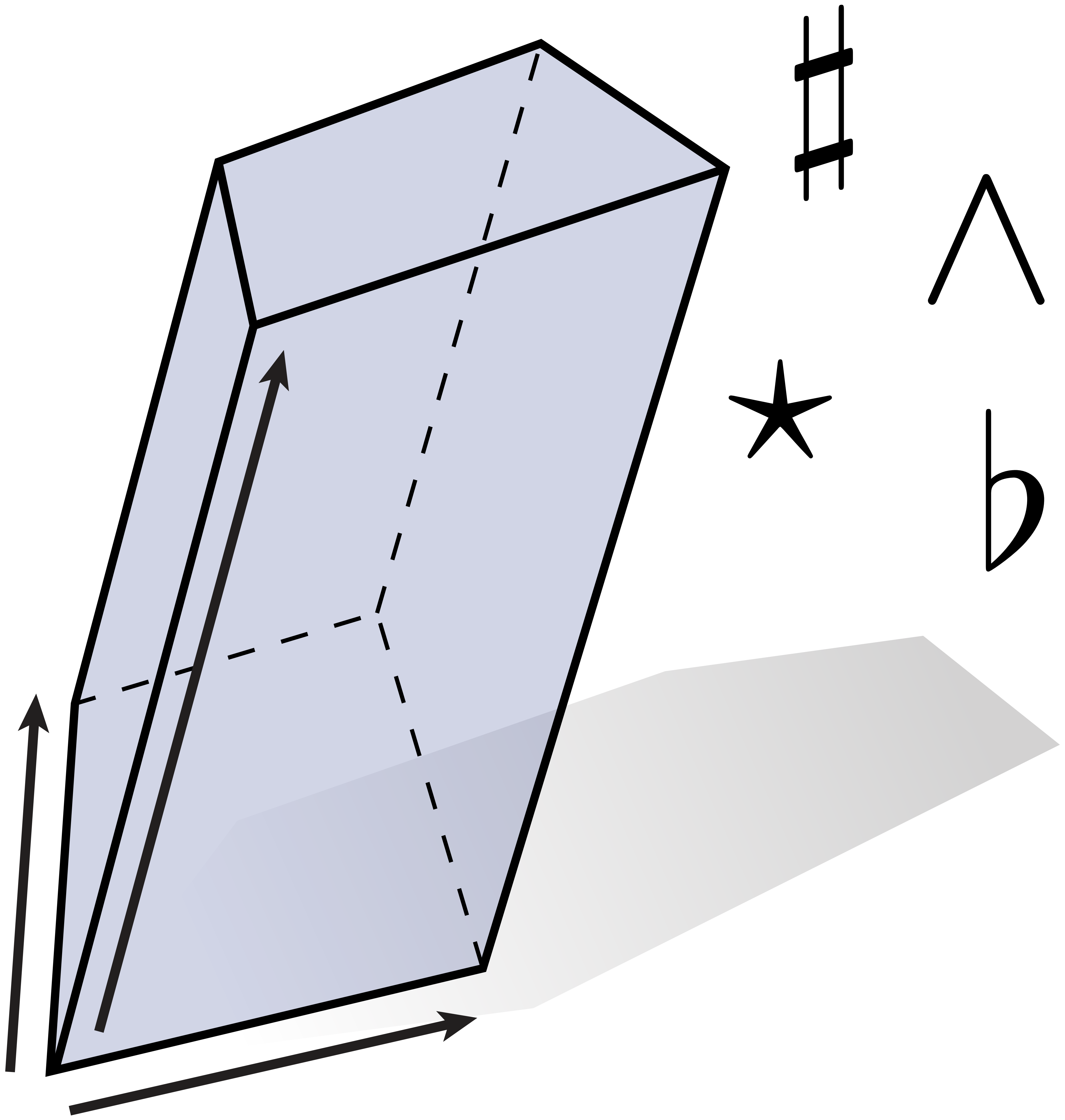
Background & Theory



Digital Geometry Processing with Discrete Exterior Calculus

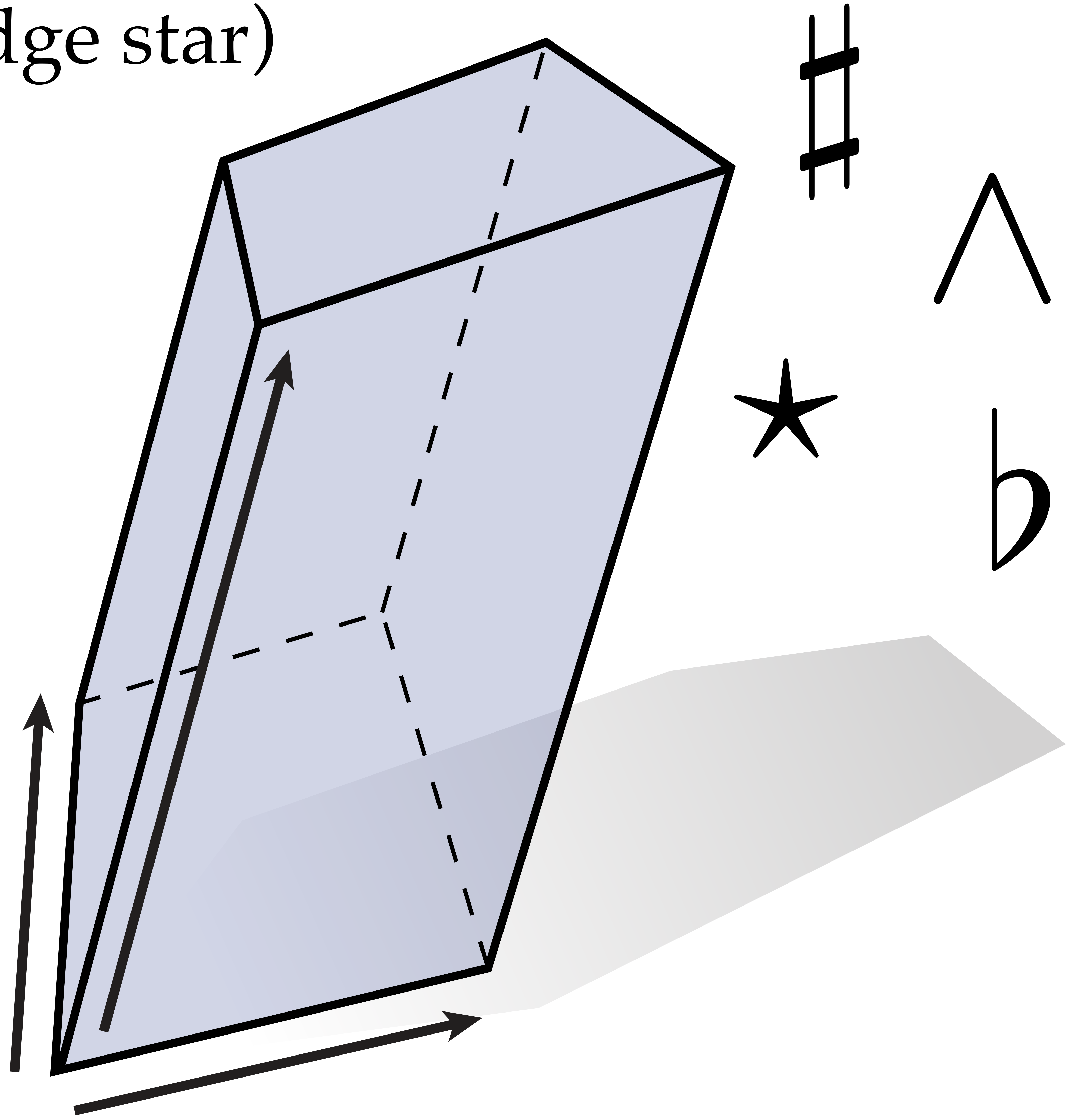
Keenan Crane • Fernando de Goes • Mathieu Desbrun • Peter Schröder

Outline



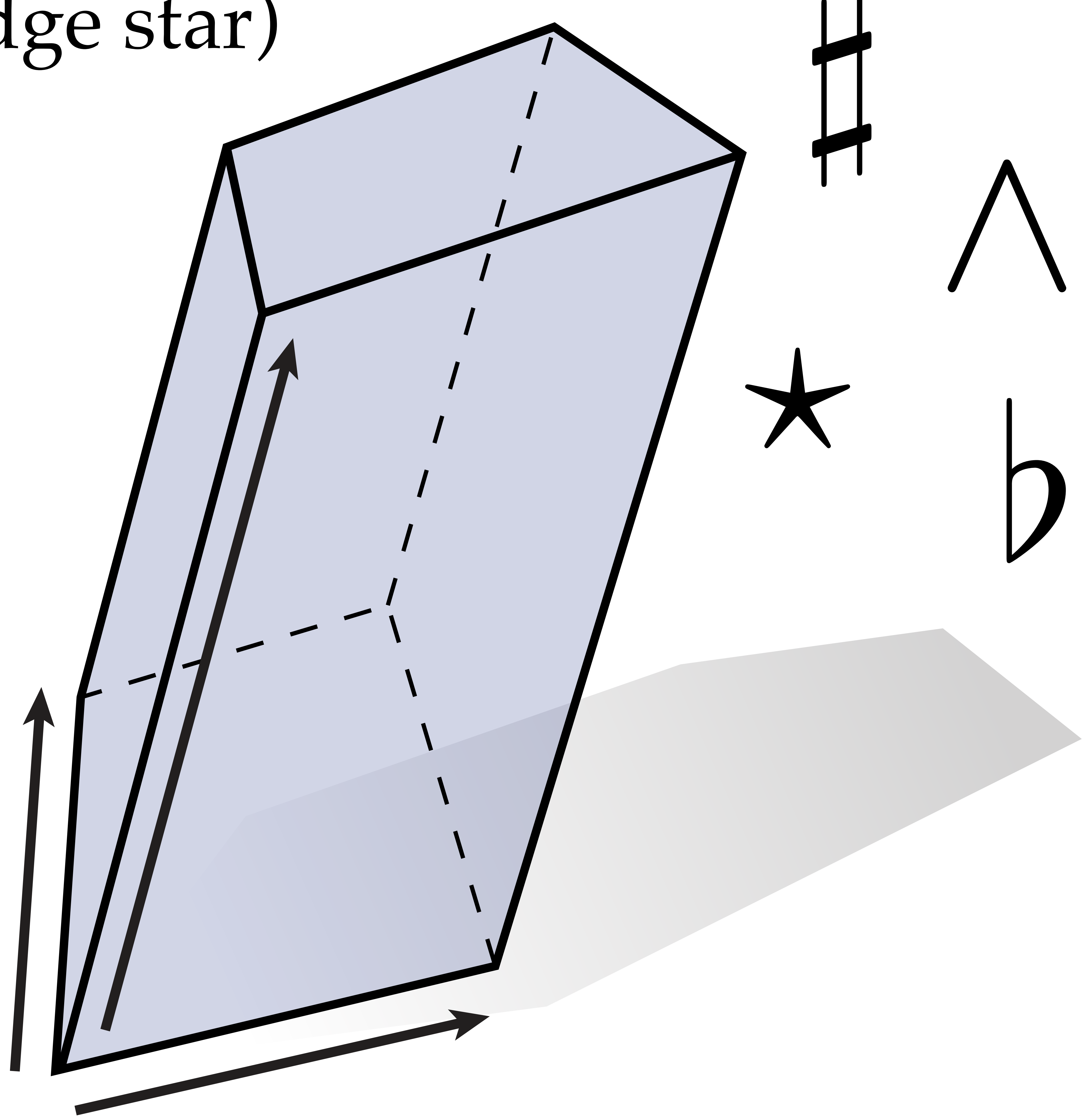
Outline

- Exterior algebra (wedge product, Hodge star)



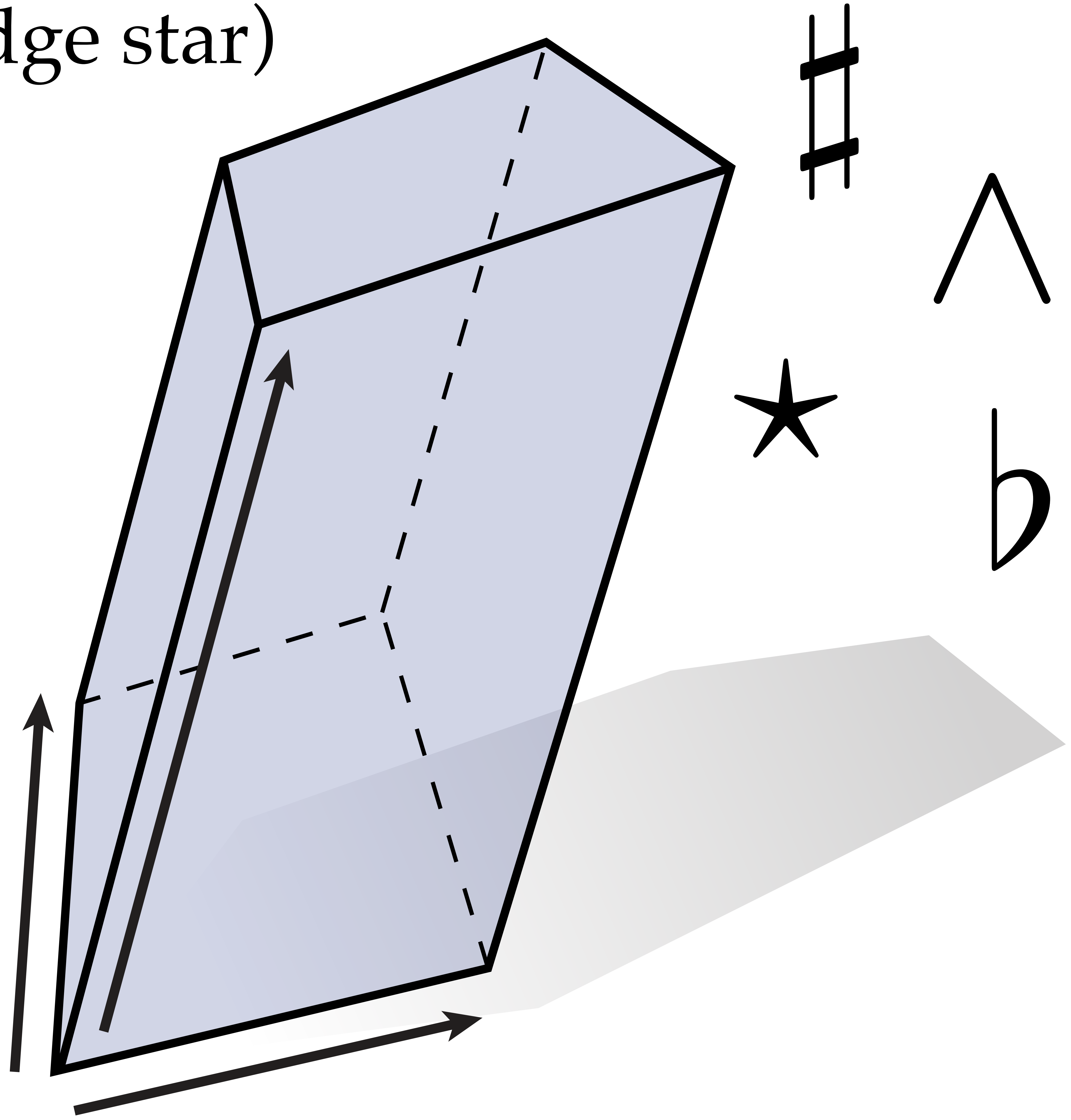
Outline

- Exterior algebra (wedge product, Hodge star)
- Exterior calculus (exterior derivative)



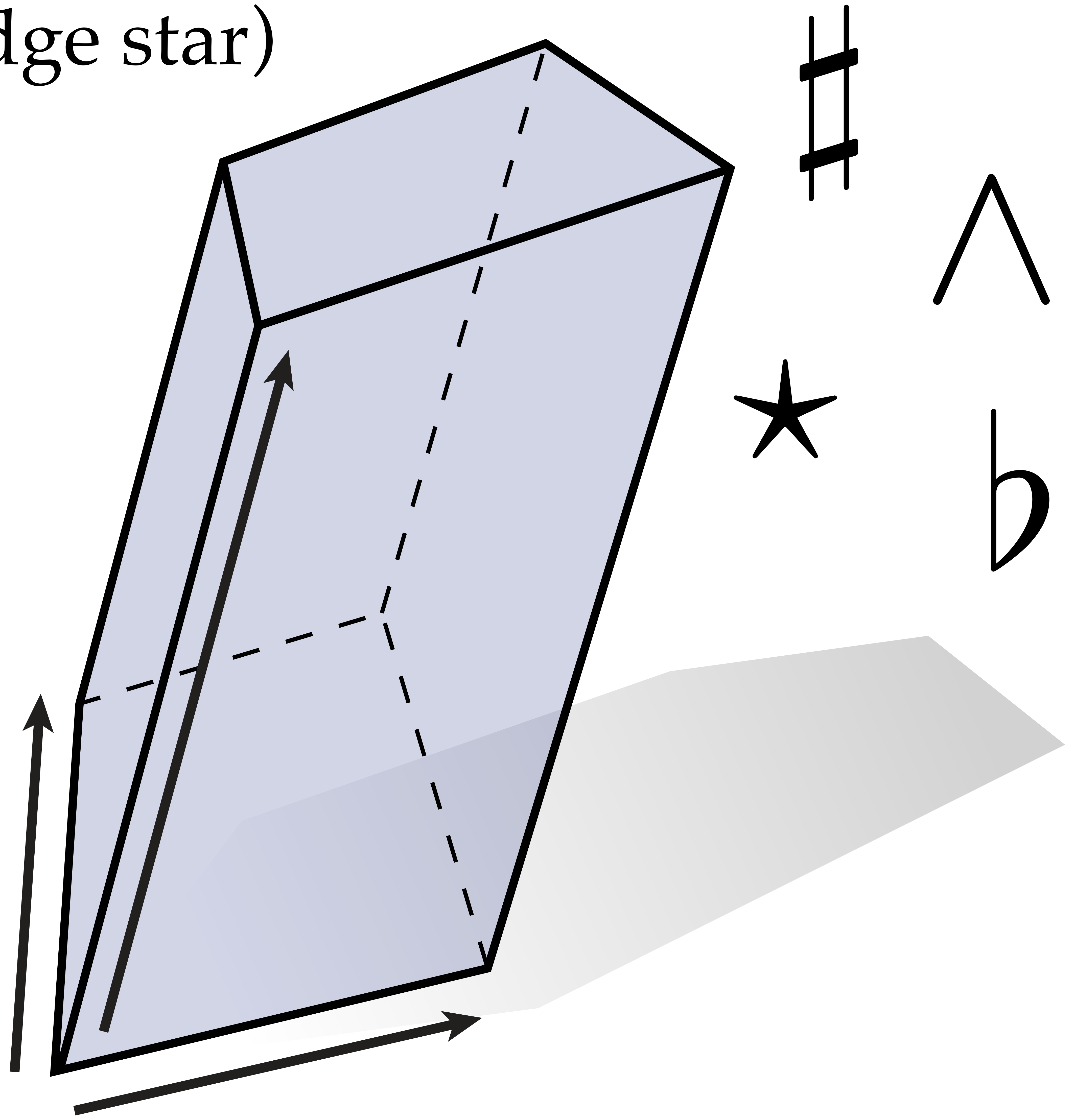
Outline

- Exterior algebra (wedge product, Hodge star)
- Exterior calculus (exterior derivative)
- Integration (Stokes' theorem, DEC)



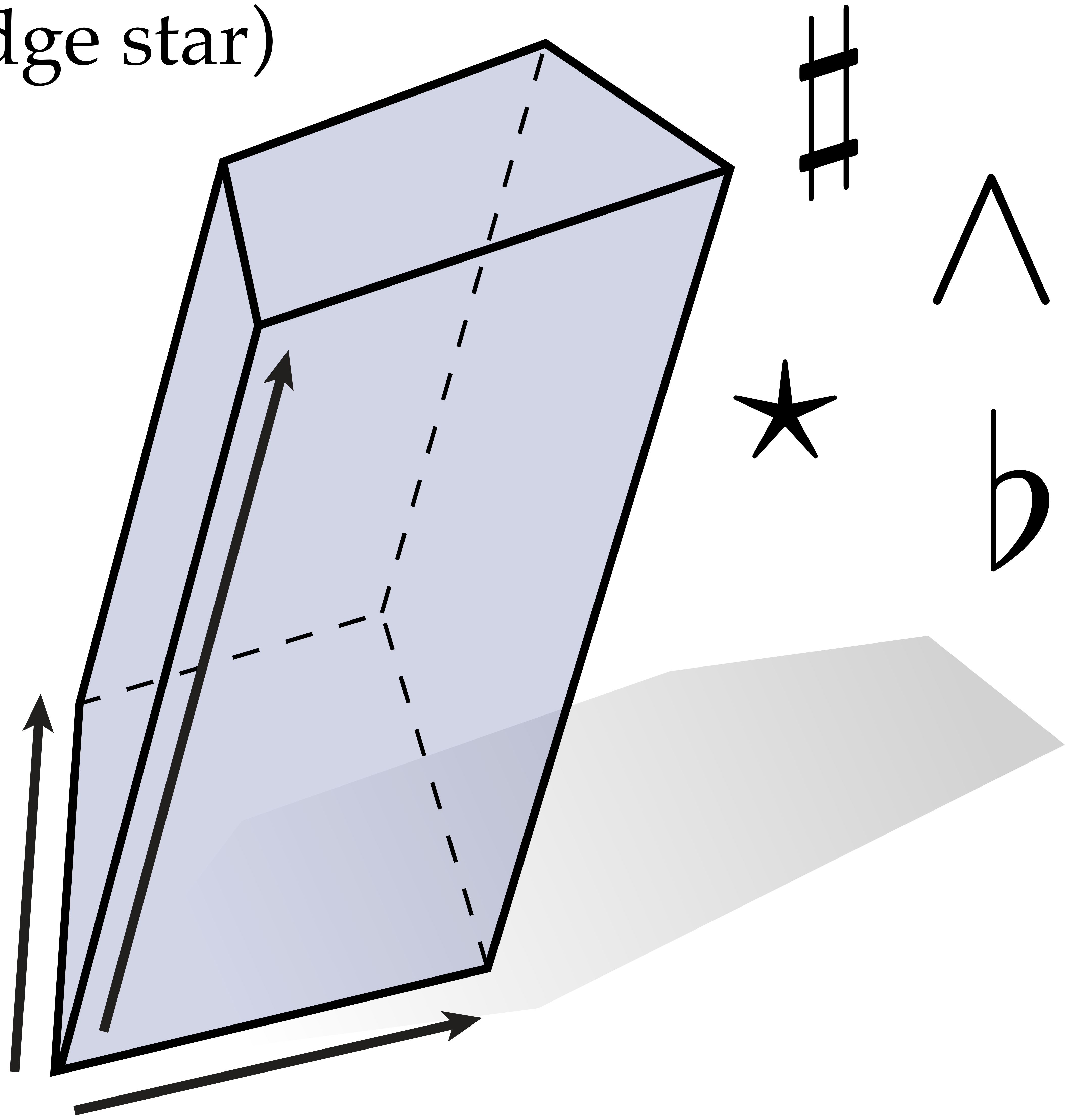
Outline

- Exterior algebra (wedge product, Hodge star)
- Exterior calculus (exterior derivative)
- Integration (Stokes' theorem, DEC)
- The Laplacian



Outline

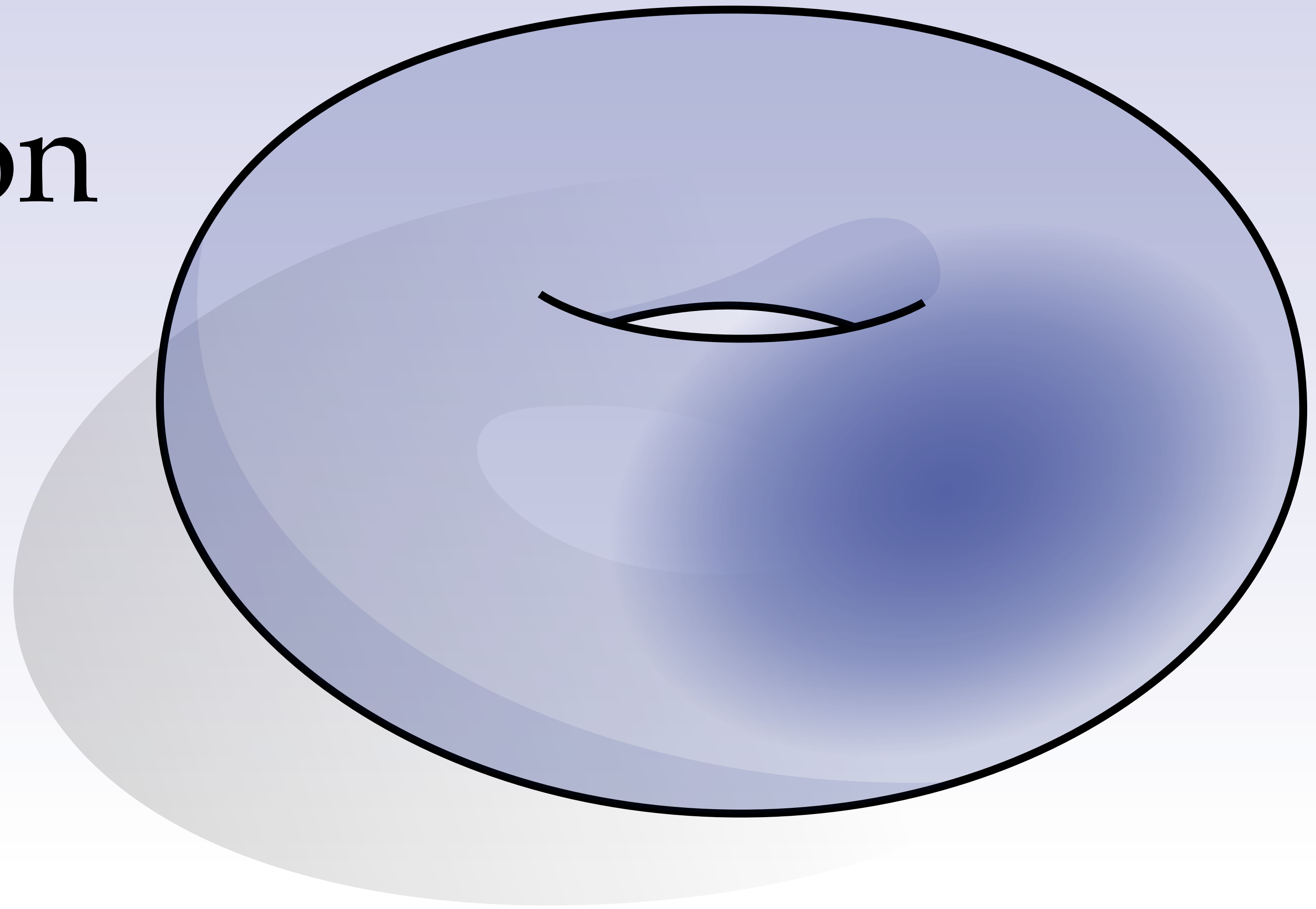
- Exterior algebra (wedge product, Hodge star)
- Exterior calculus (exterior derivative)
- Integration (Stokes' theorem, DEC)
- The Laplacian
- Homology & cohomology



Basic Computational Tools

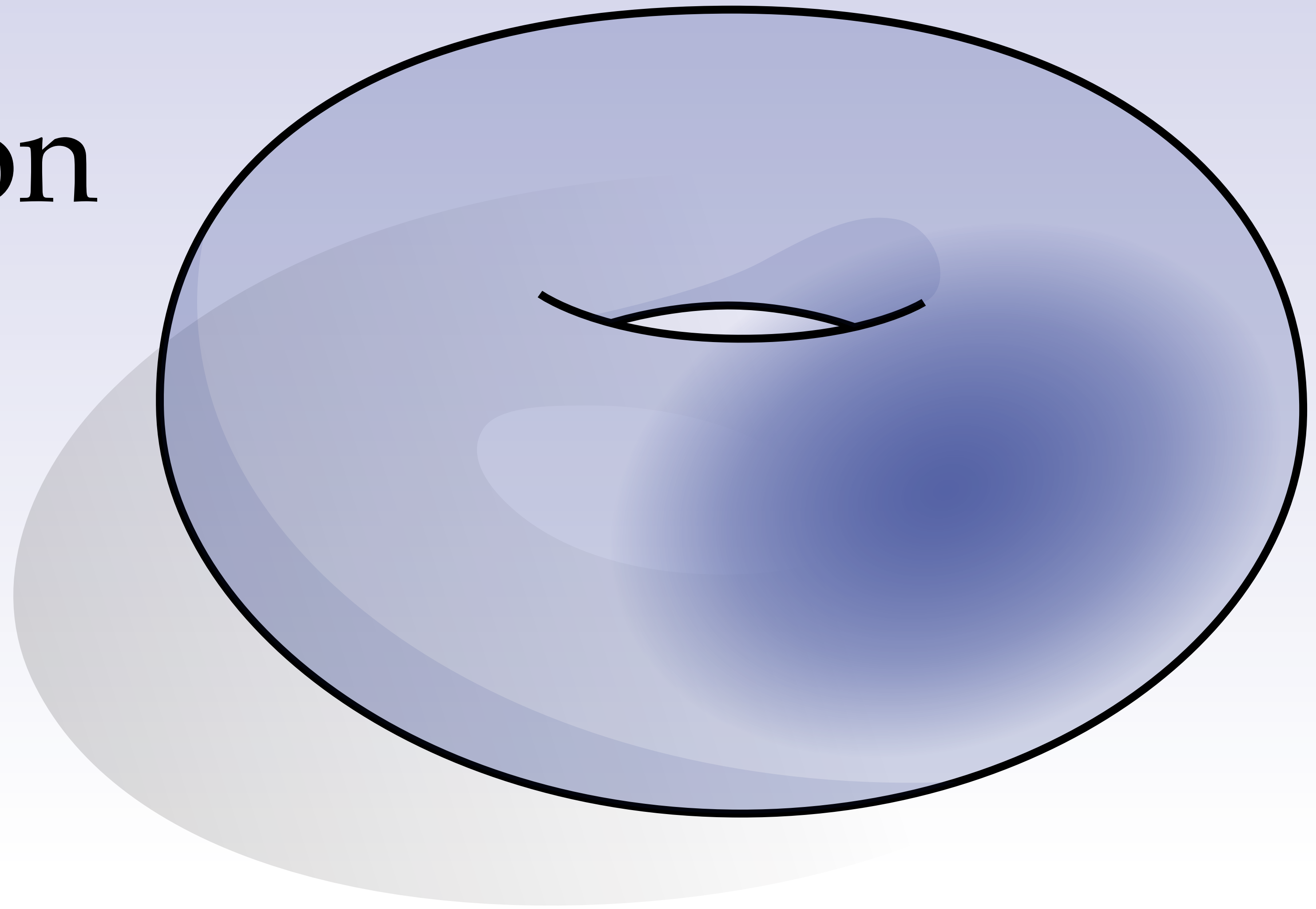
Basic Computational Tools

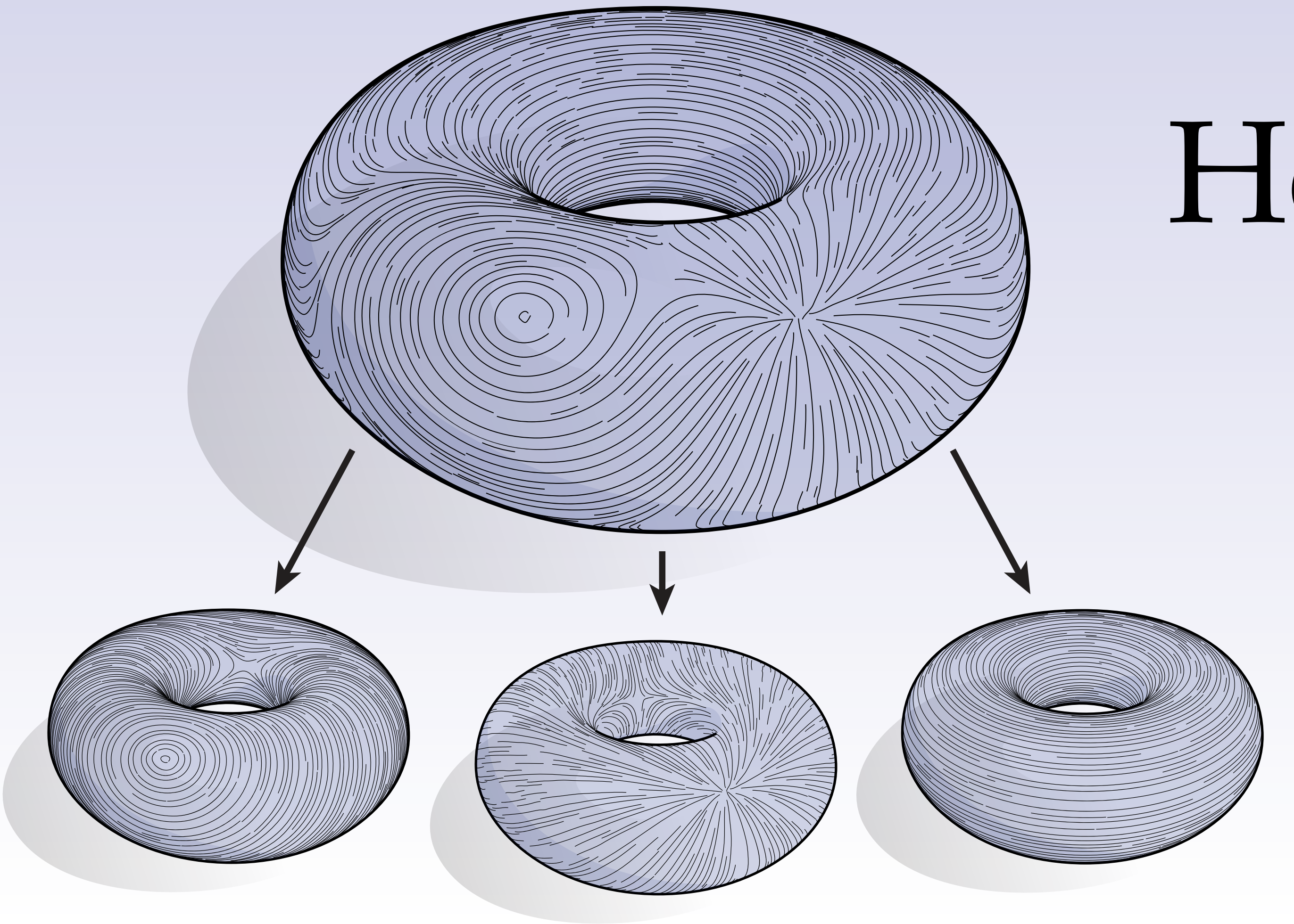
Poisson



Basic Computational Tools

Poisson

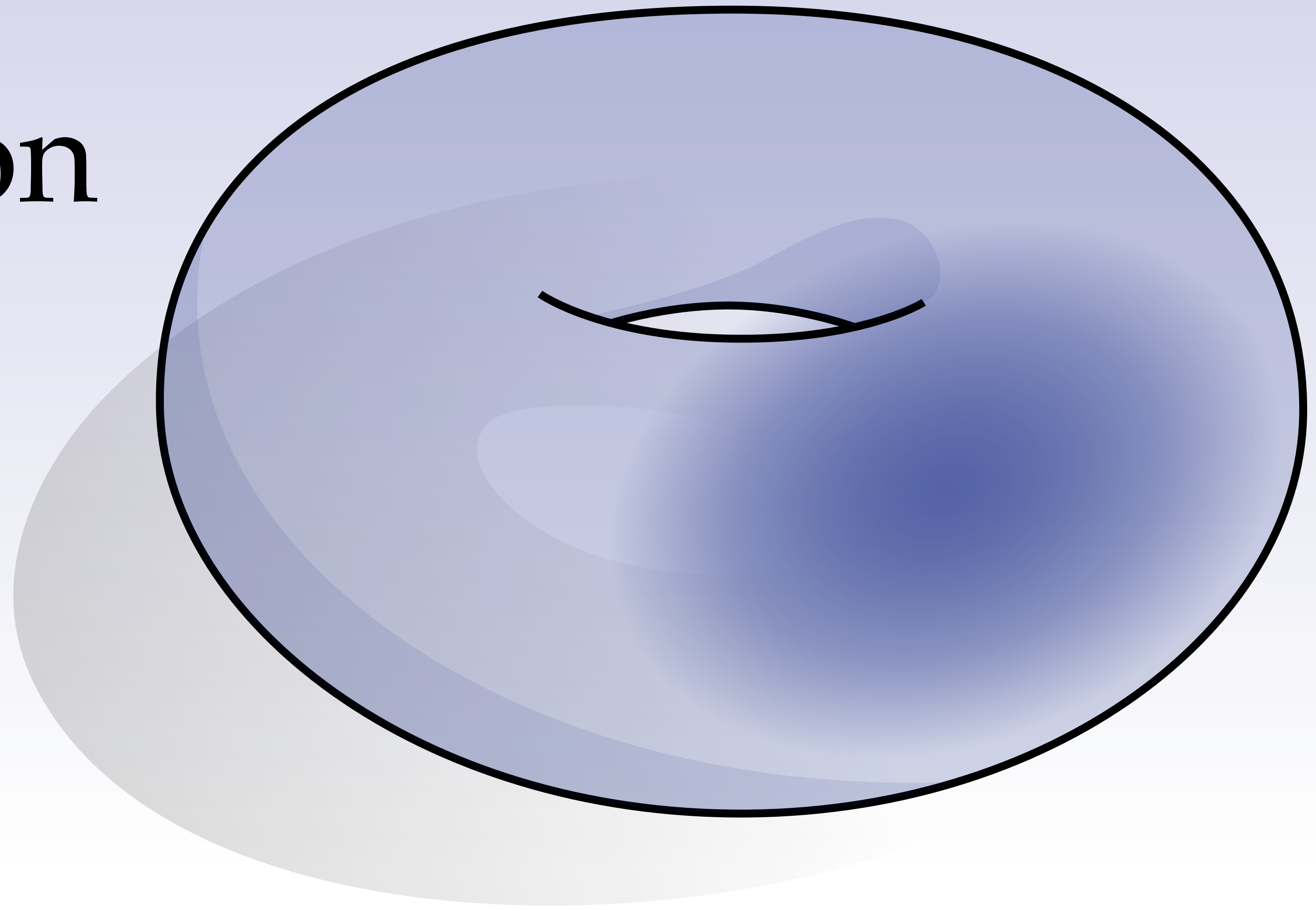




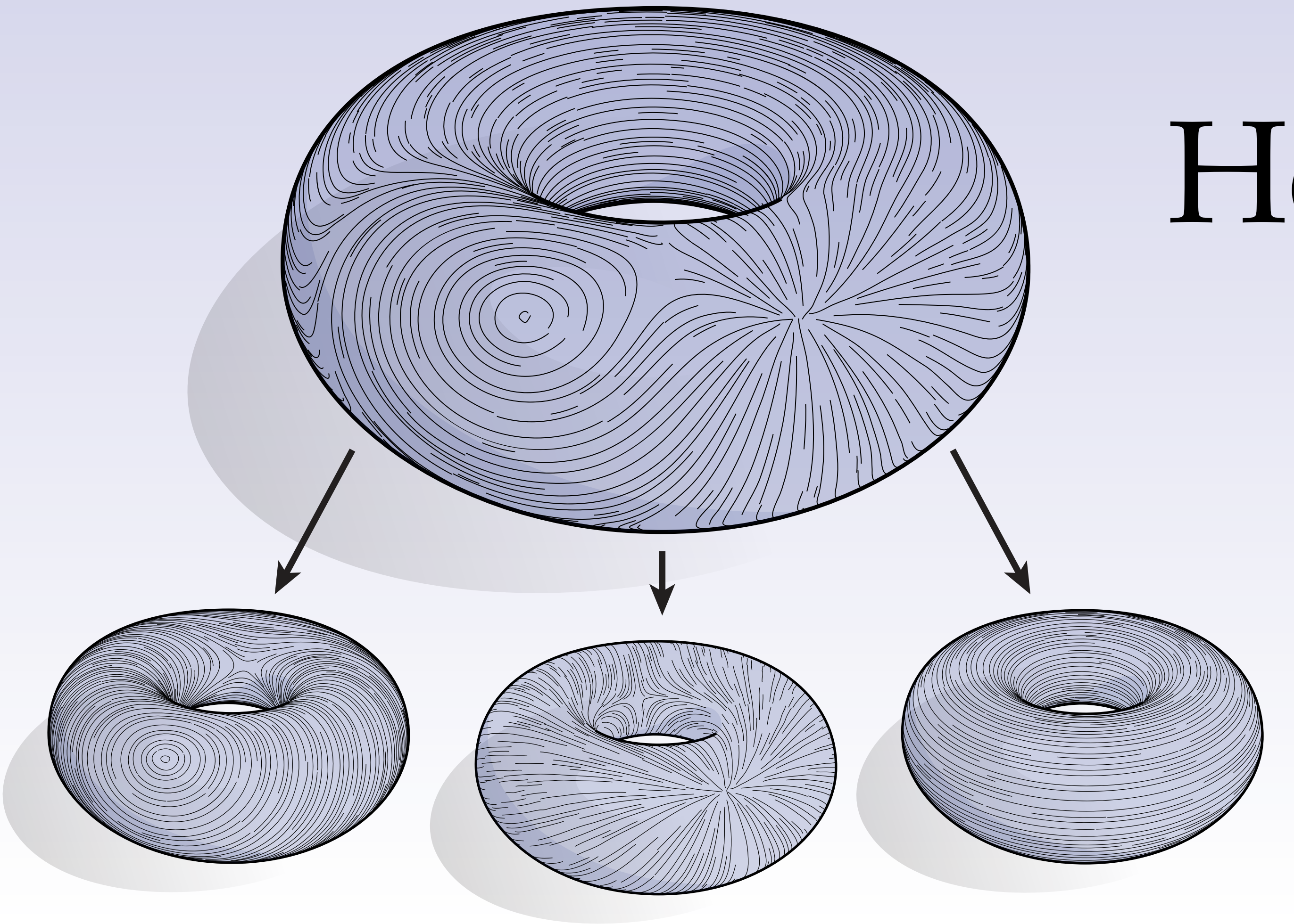
Helmholtz-
Hodge

Basic Computational Tools

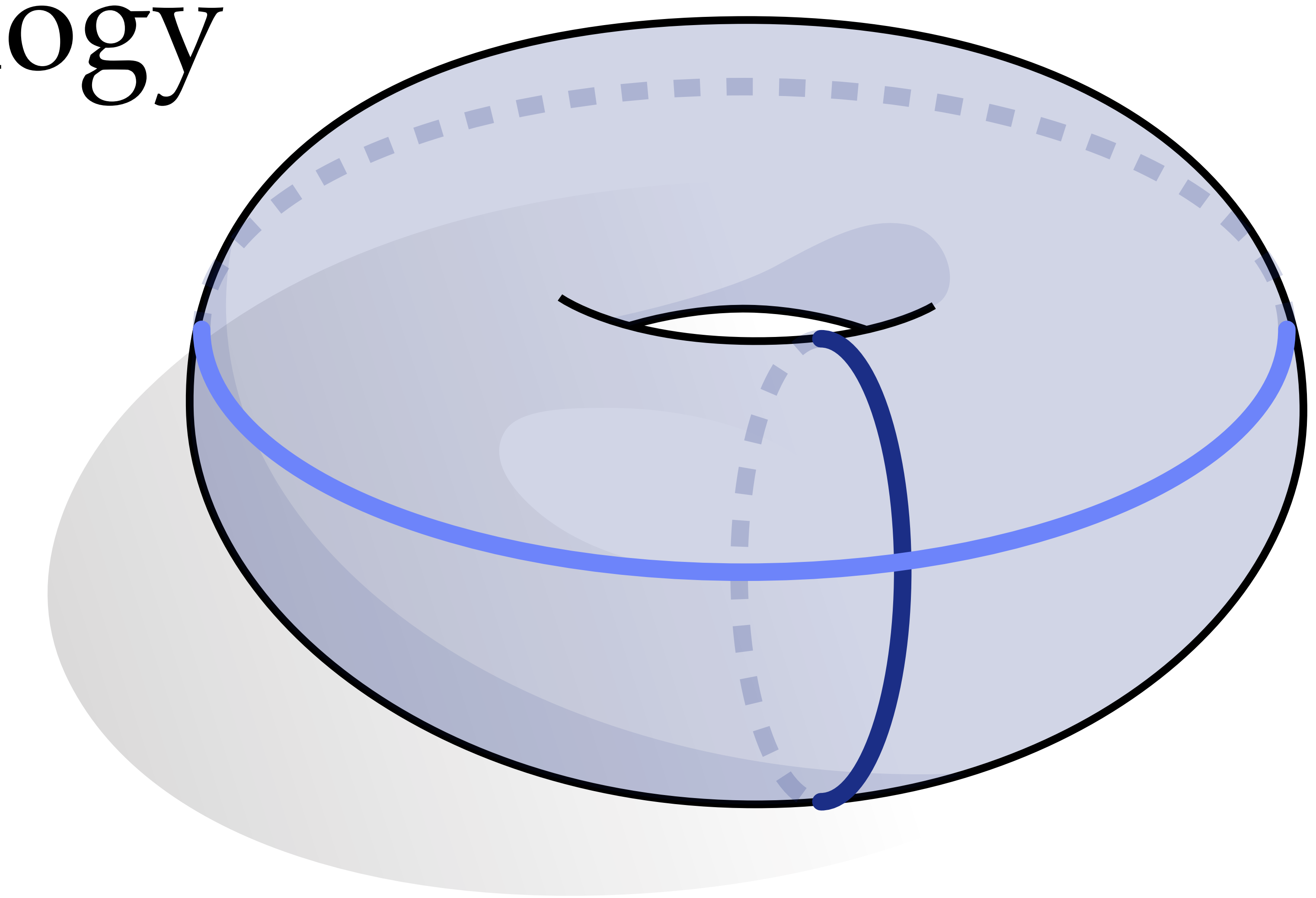
Poisson



Helmholtz-
Hodge

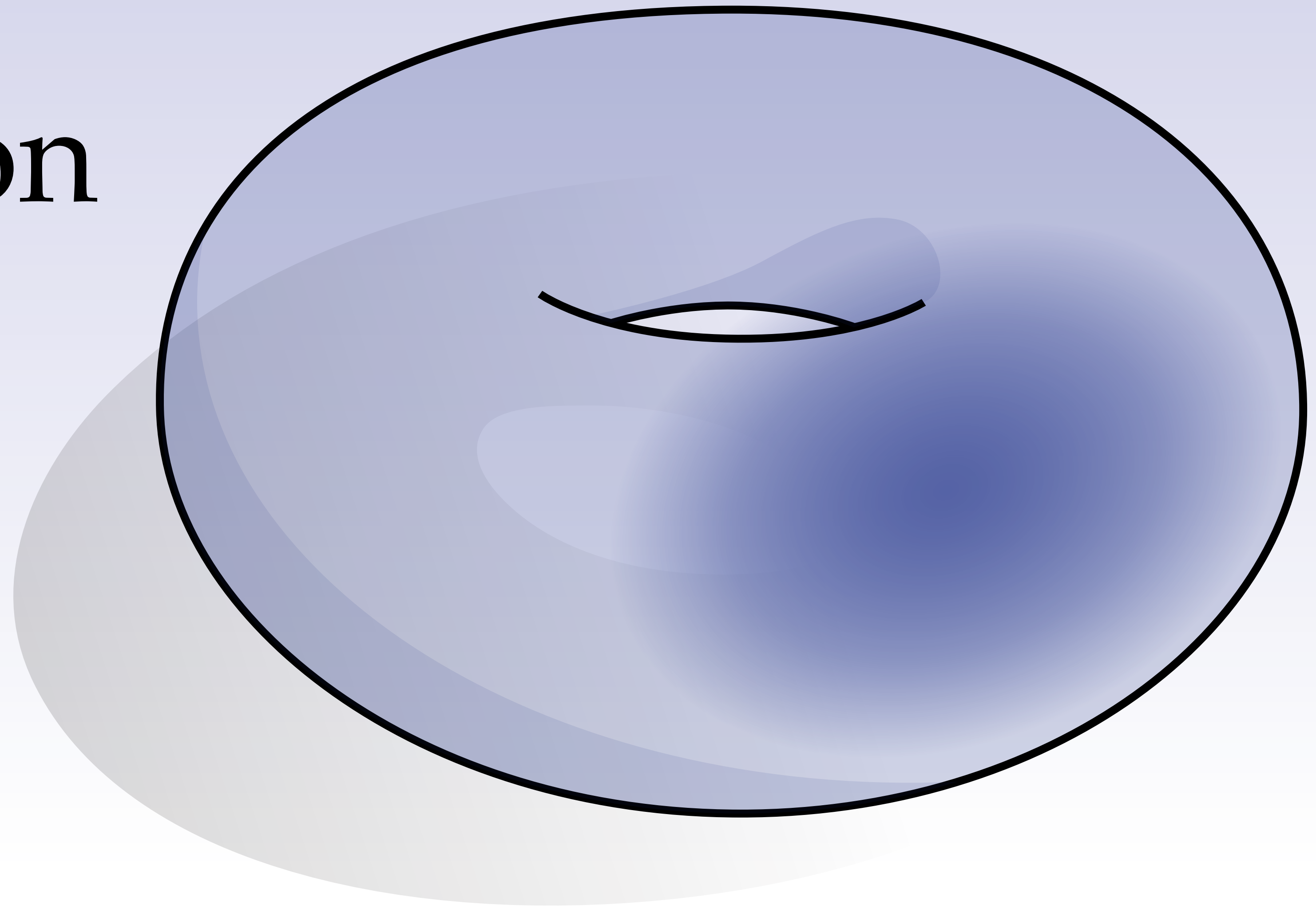


homology

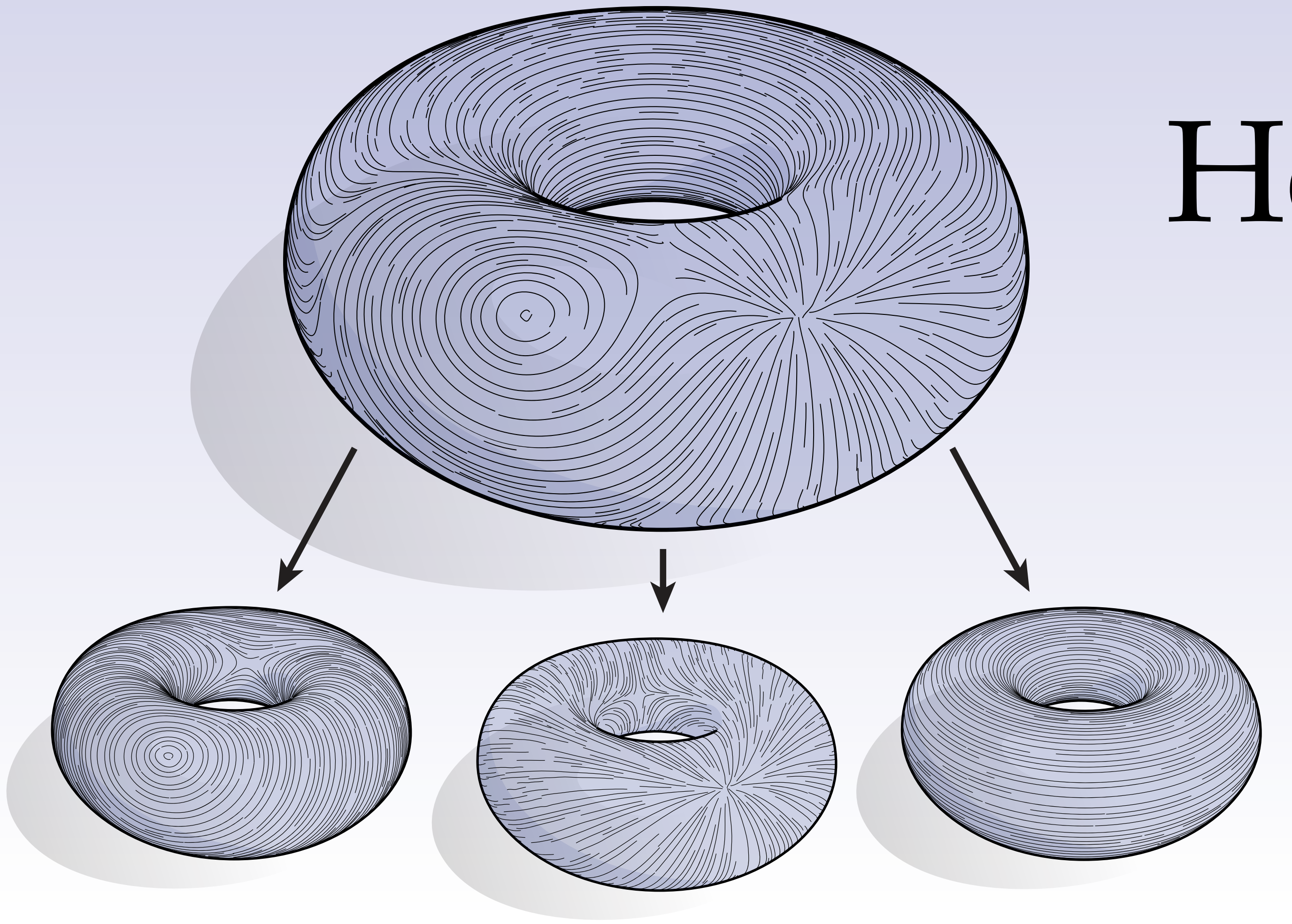


Basic Computational Tools

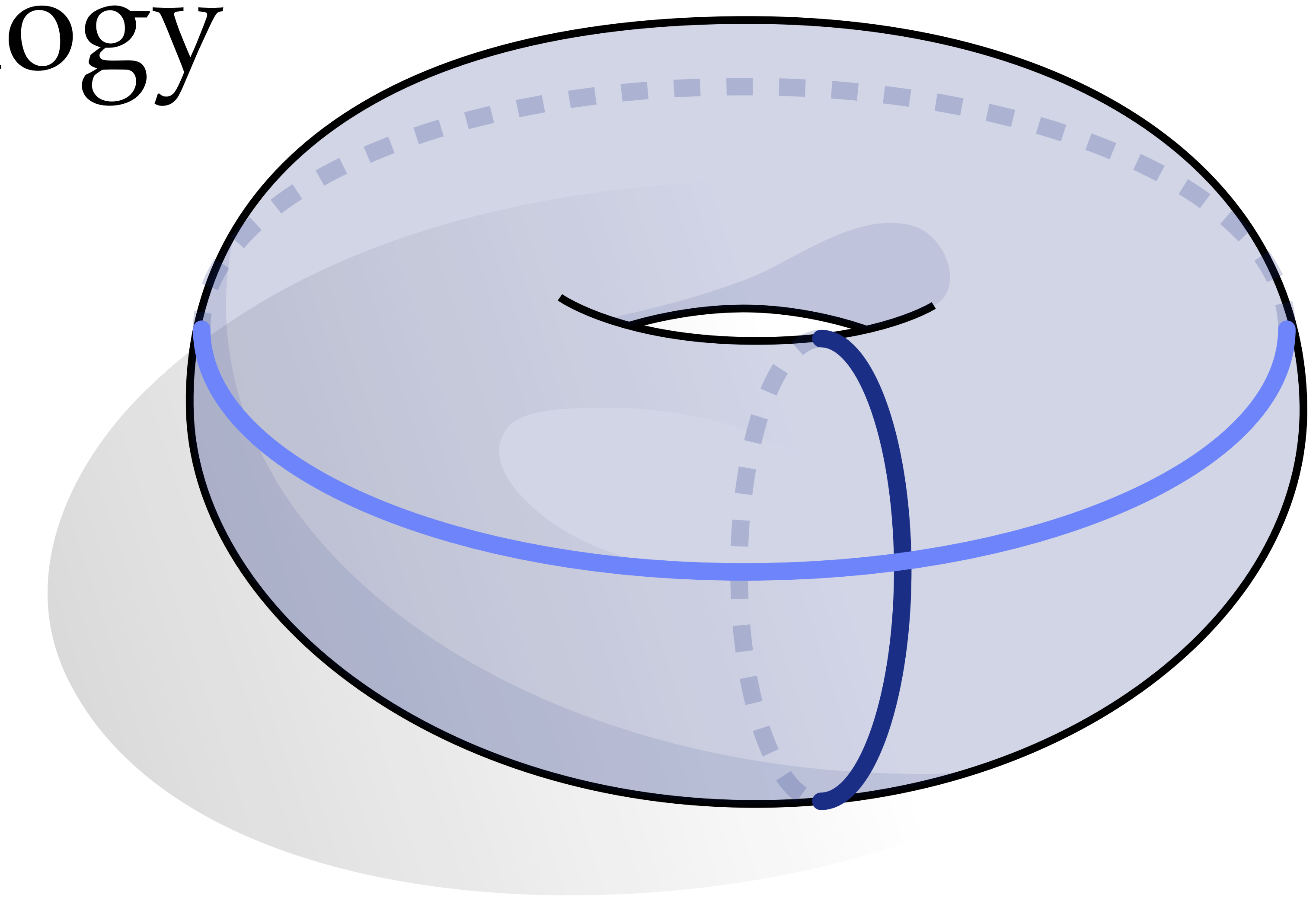
Poisson



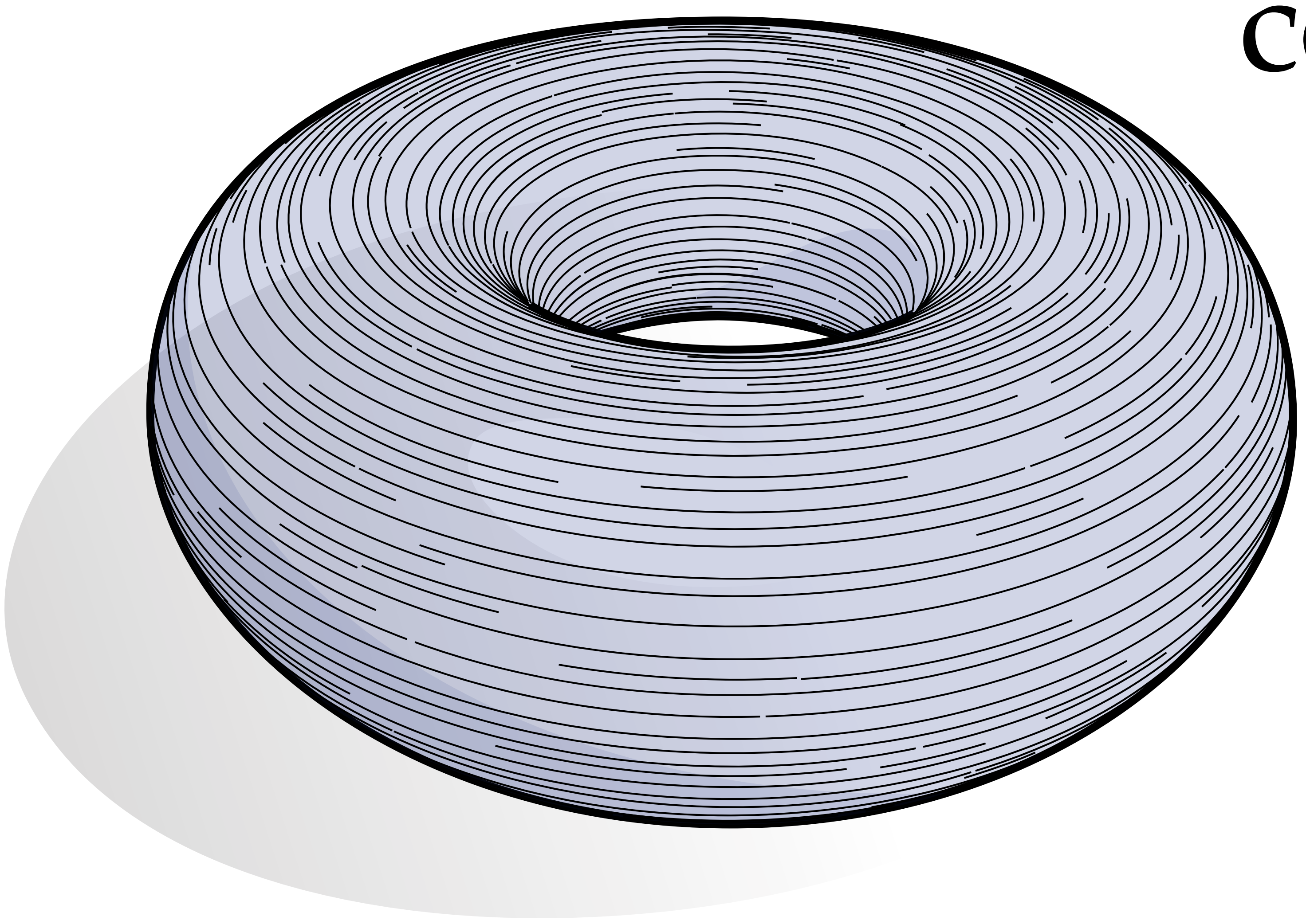
Helmholtz-Hodge



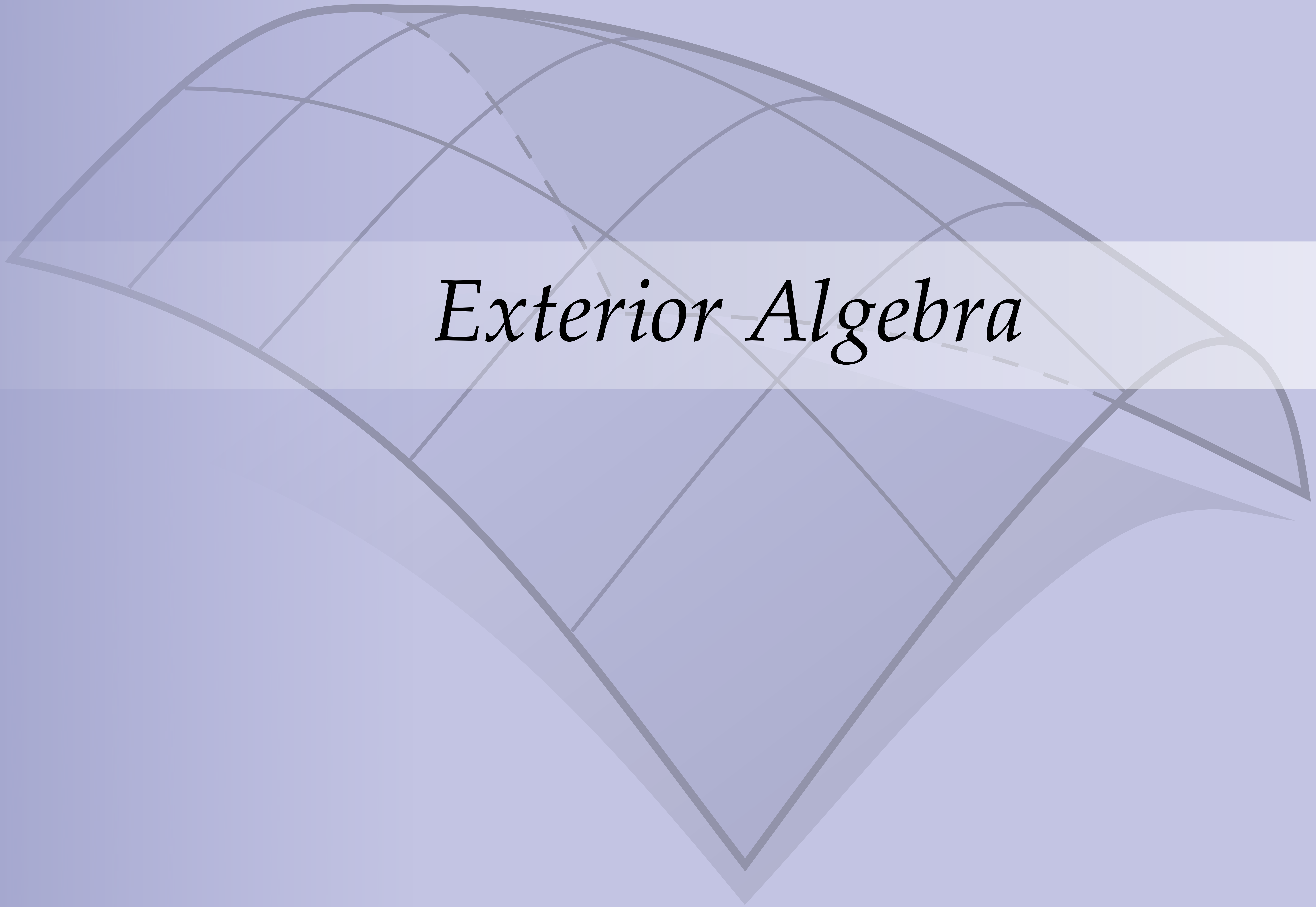
homology



cohomology



Exterior Algebra



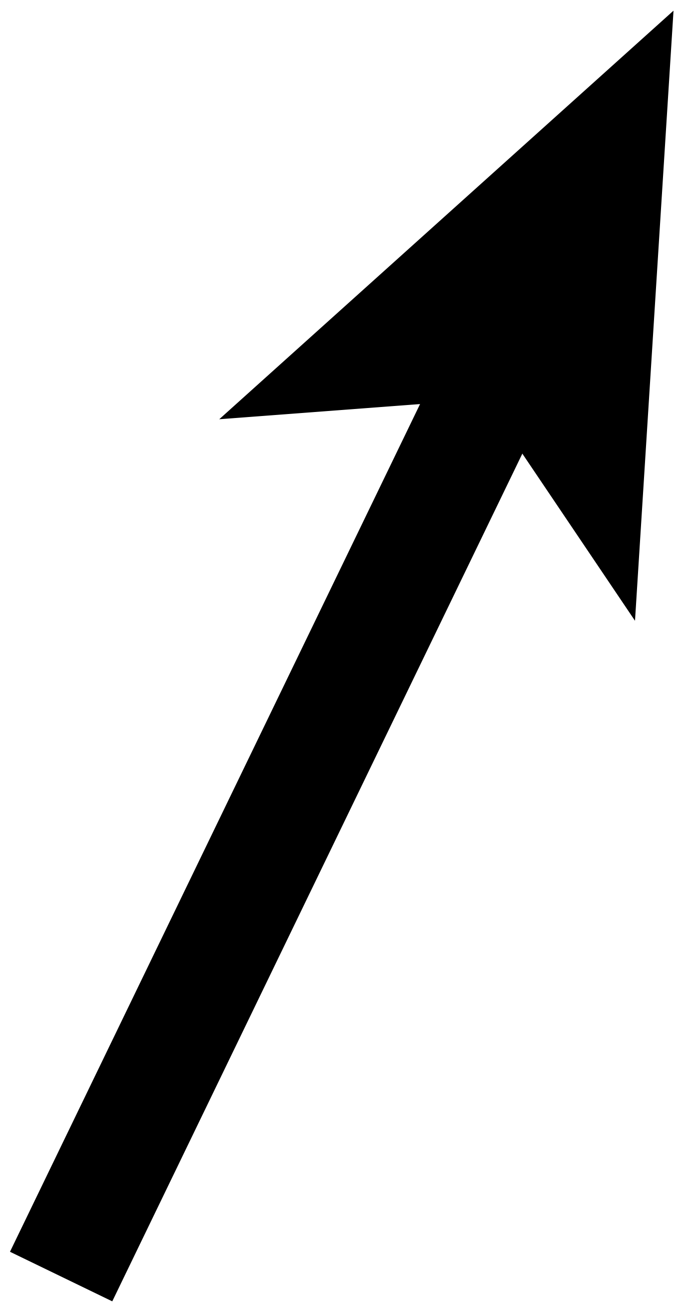
Review: Vector Spaces

Review: Vector Spaces

- What is a vector? *(Geometrically?)*

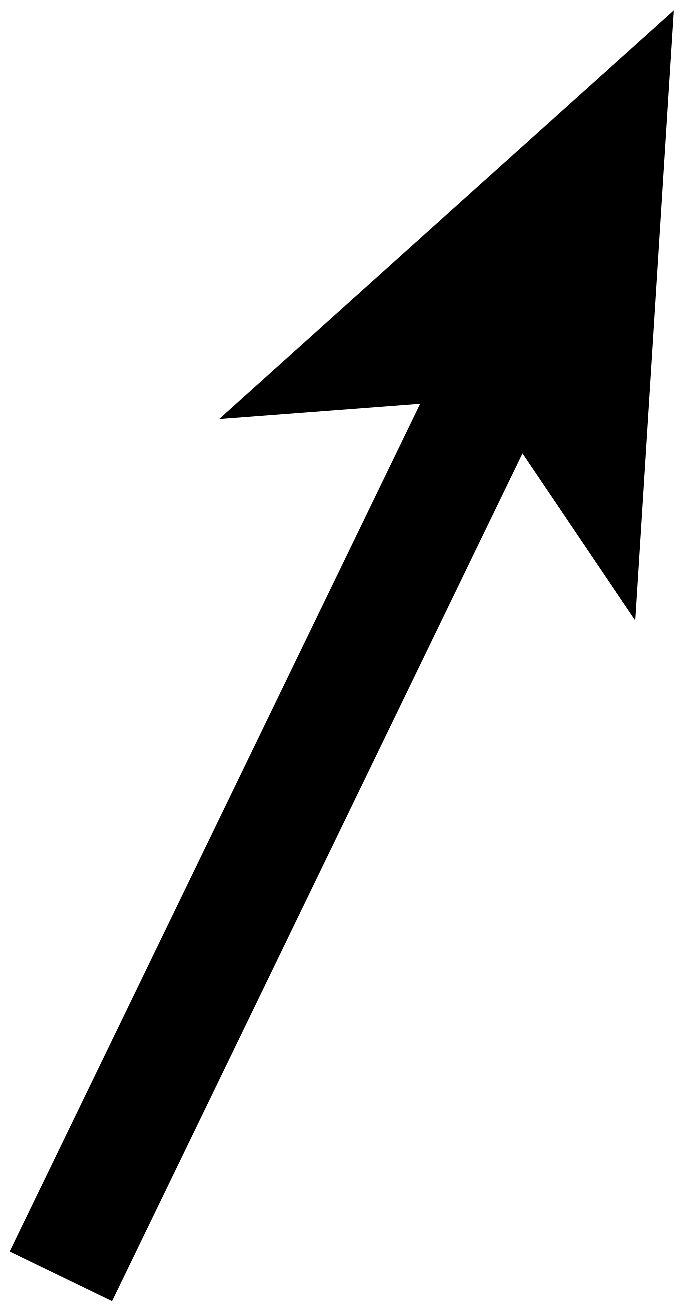
Review: Vector Spaces

- What is a vector? *(Geometrically?)*



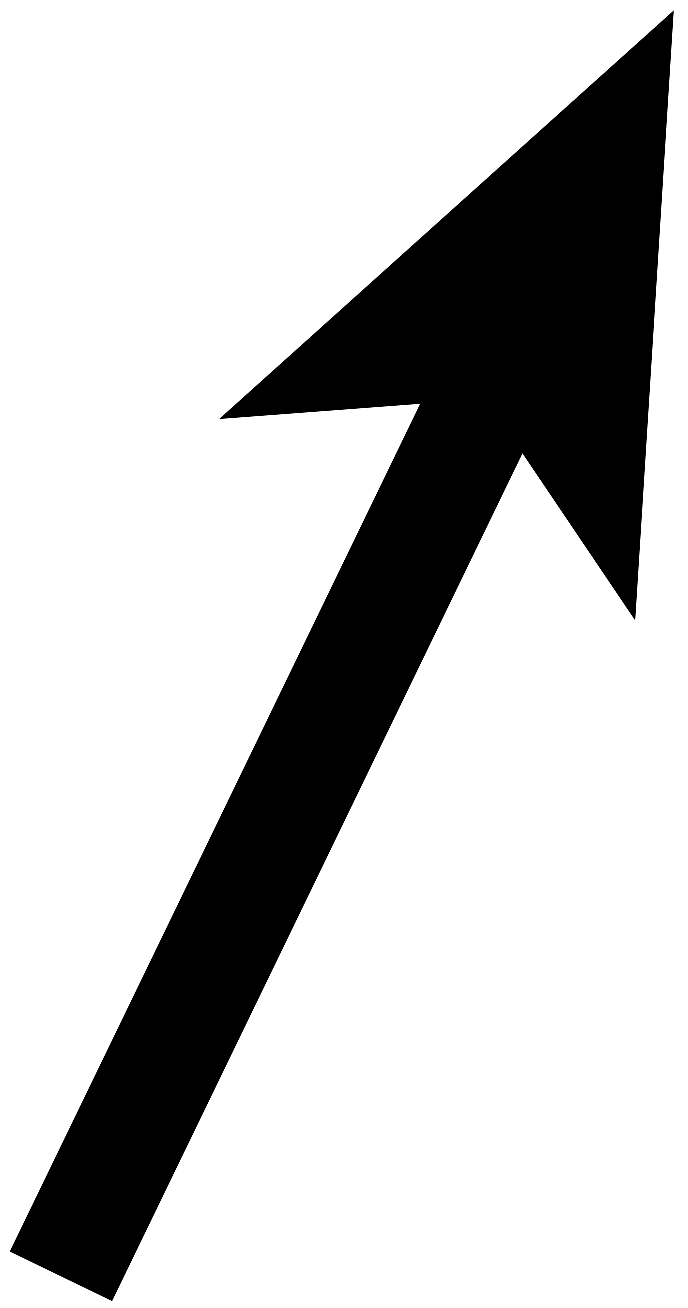
Review: Vector Spaces

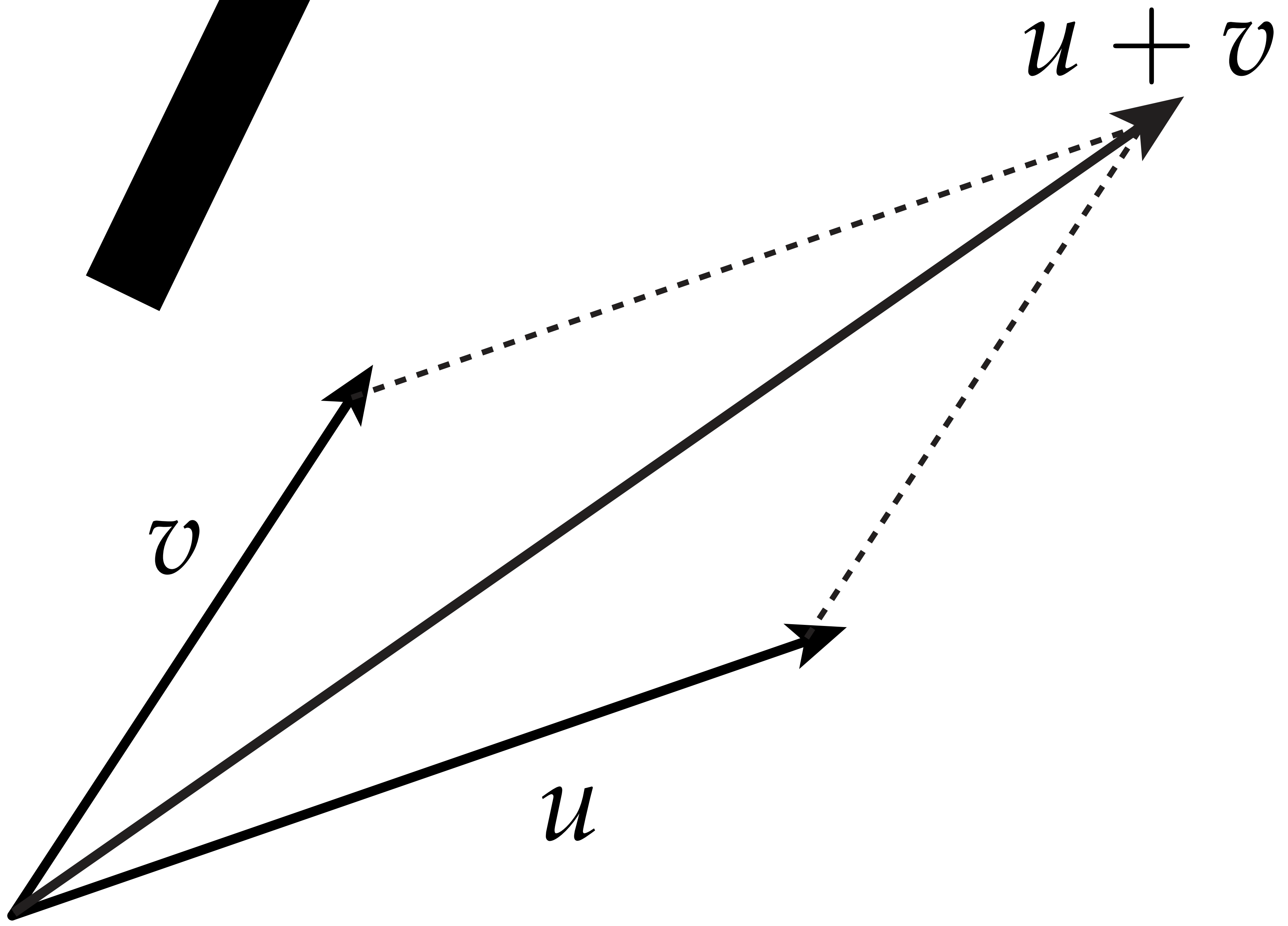
- What is a vector? *(Geometrically?)*
- What is a vector *space*?



Review: Vector Spaces

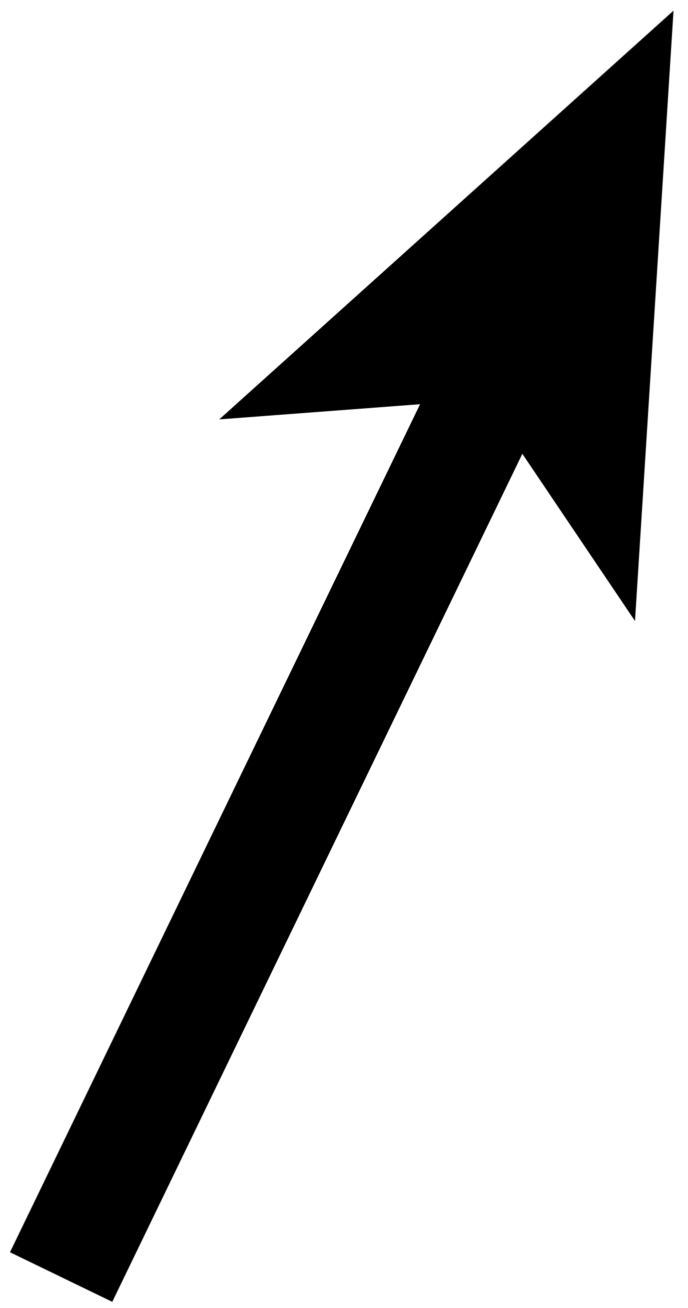
- What is a vector? *(Geometrically?)*
- What is a vector *space*?

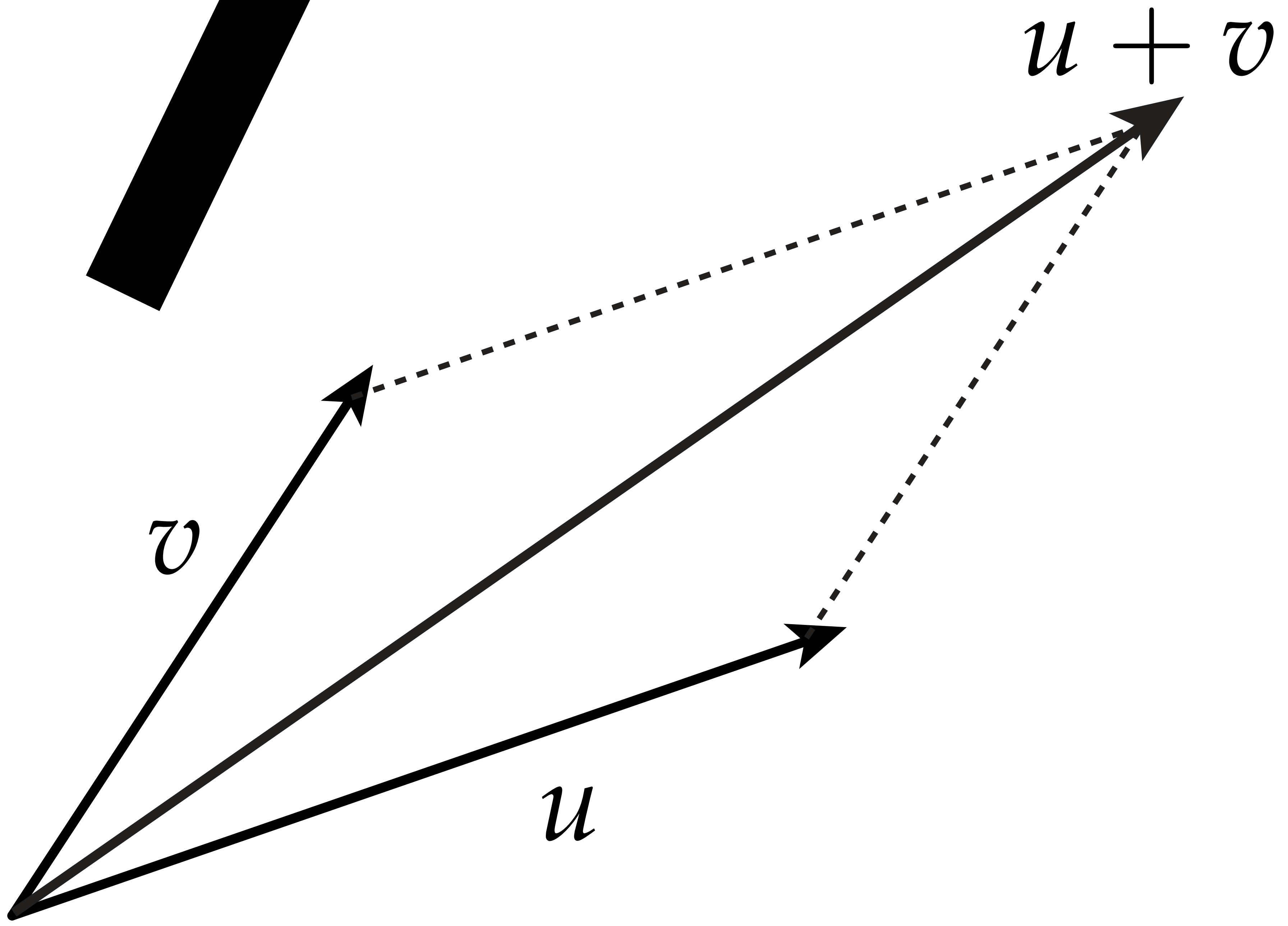




Review: Vector Spaces

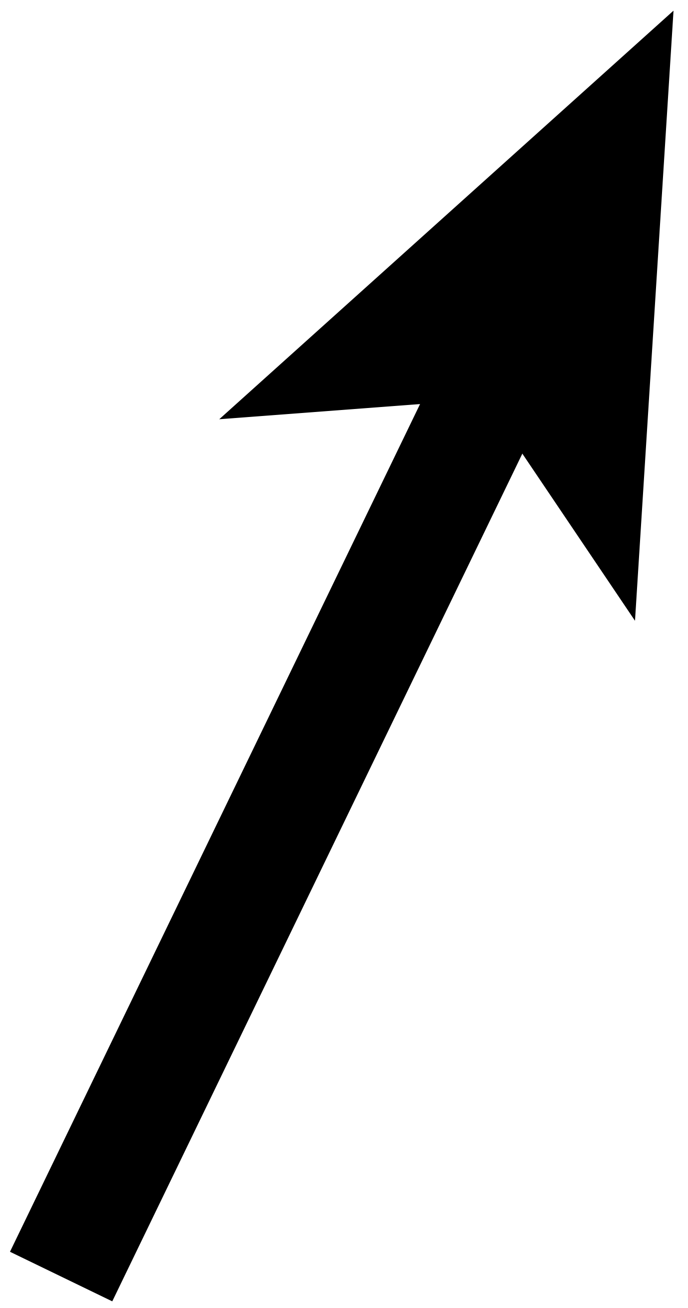
- What is a vector? *(Geometrically?)*
- What is a vector *space*?
- What is the *span*?

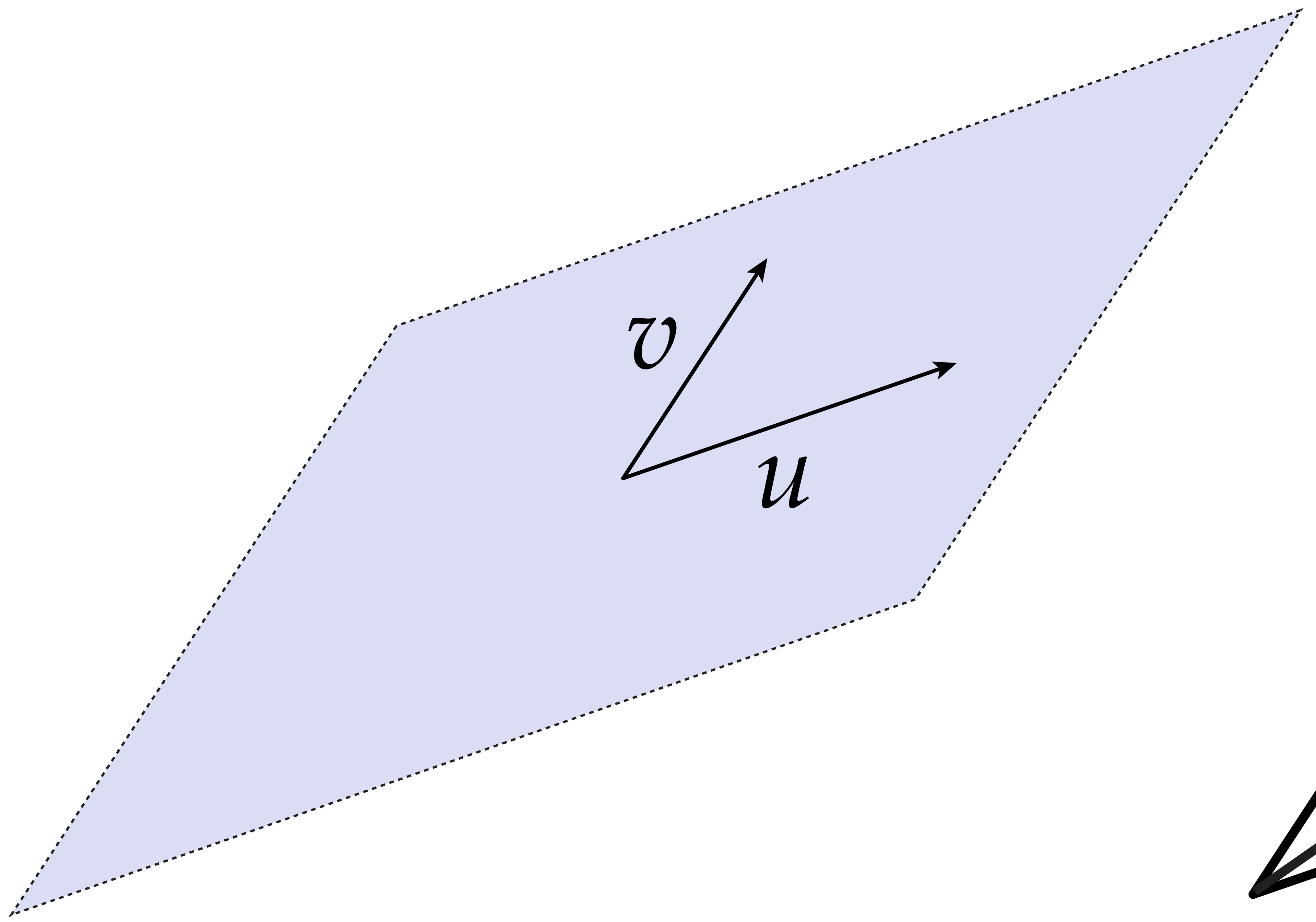


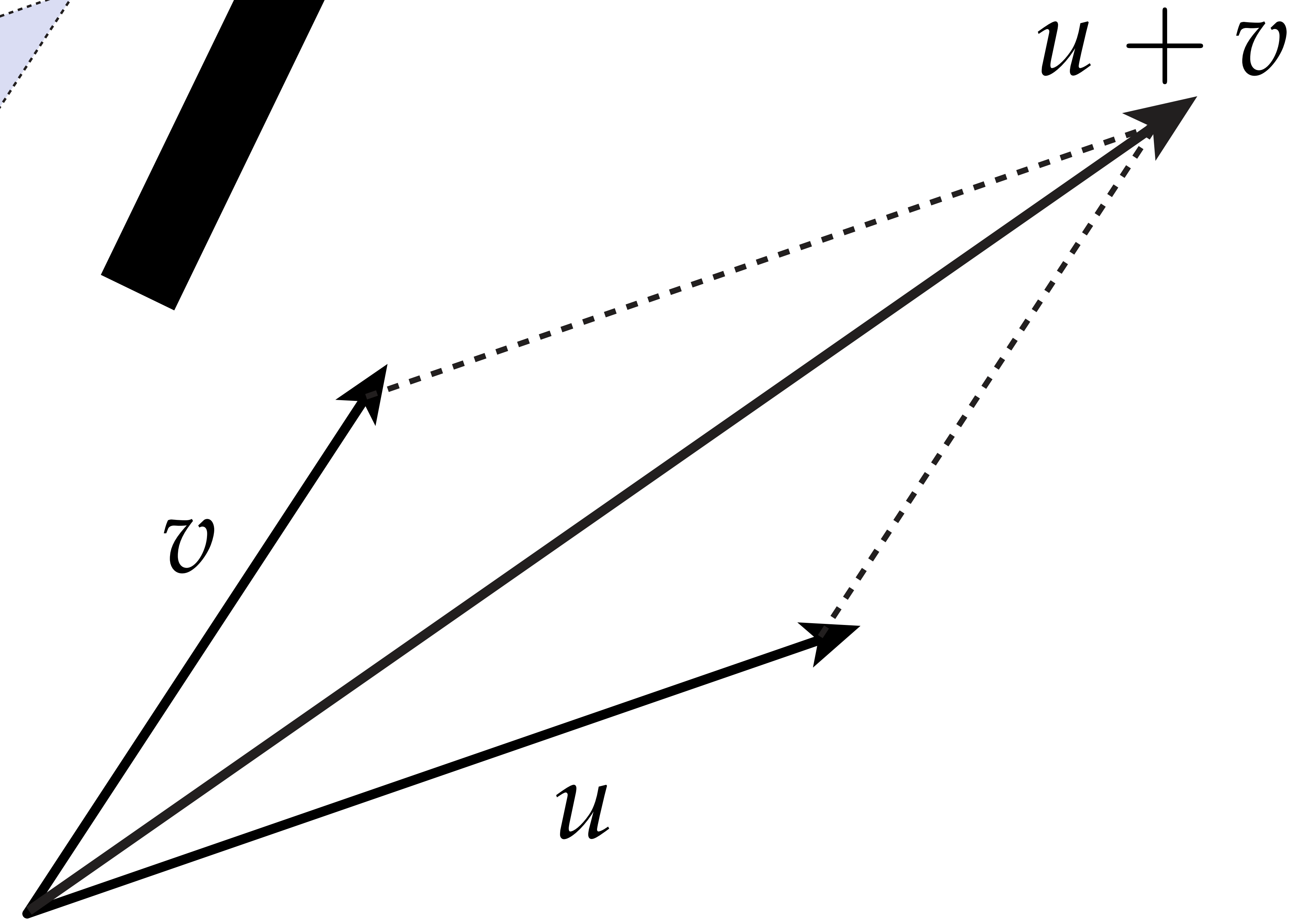


Review: Vector Spaces

- What is a vector? *(Geometrically?)*
- What is a vector *space*?
- What is the *span*?

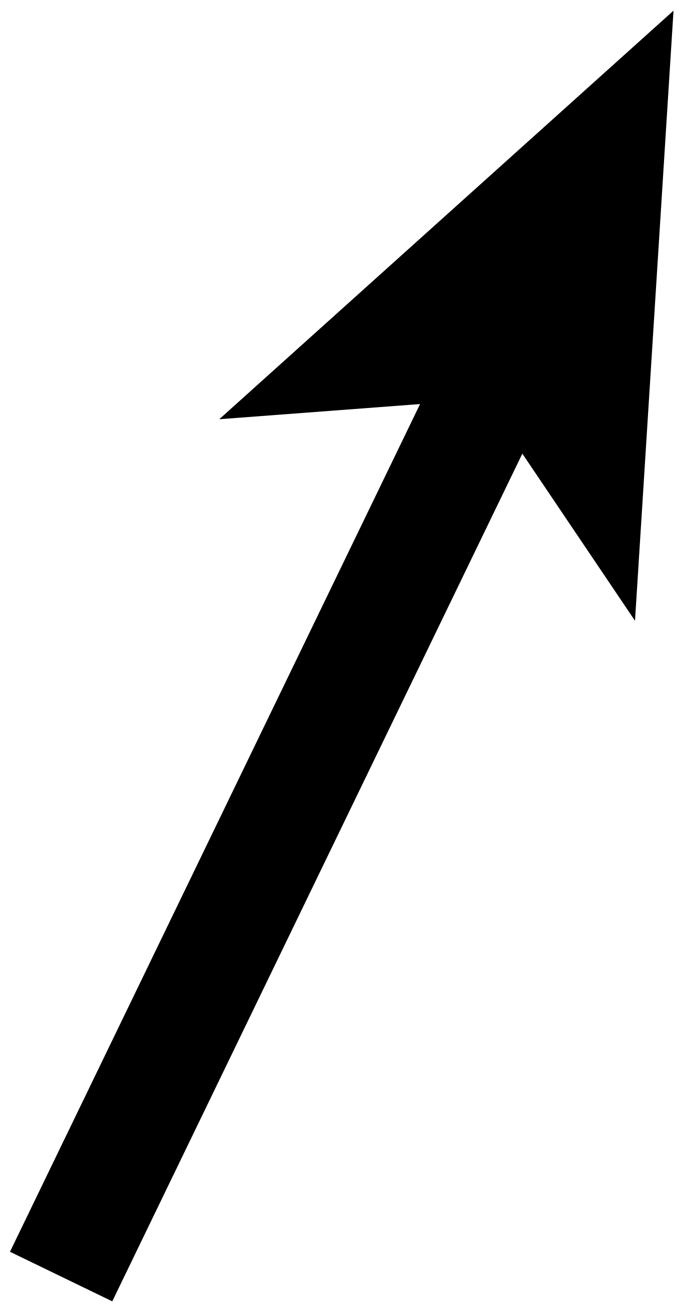


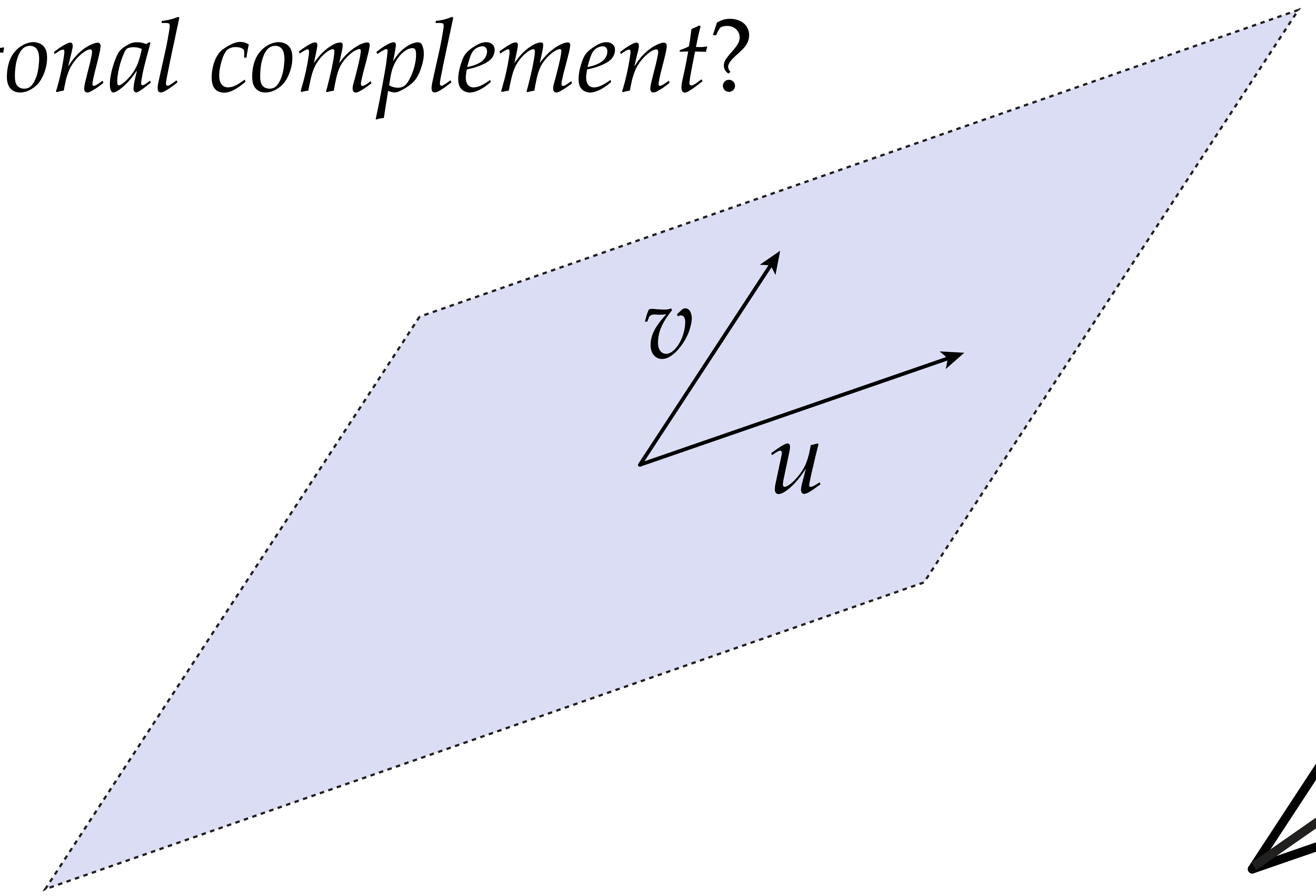


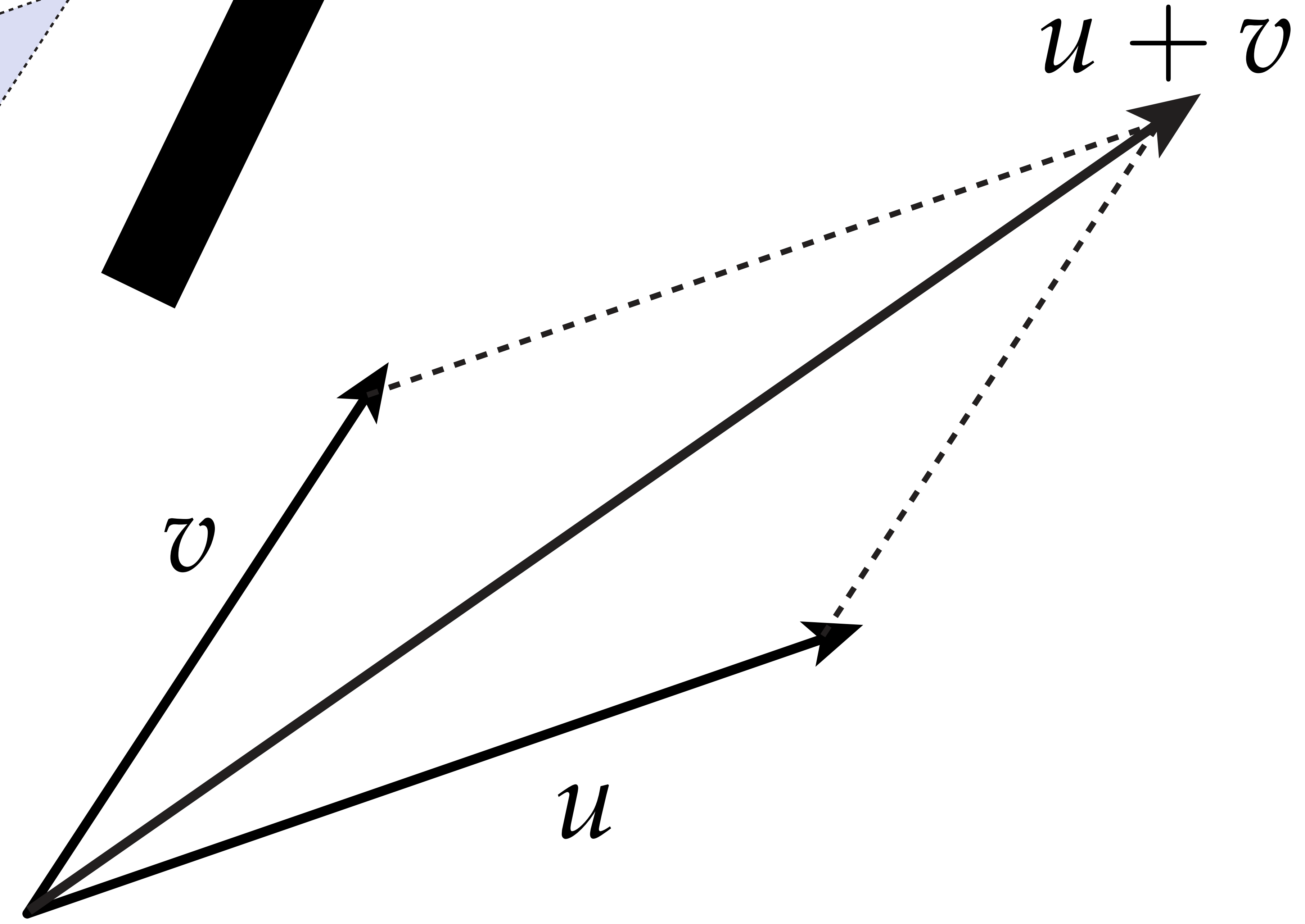


Review: Vector Spaces

- What is a vector? *(Geometrically?)*
- What is a vector *space*?
- What is the *span*?
- What is an *orthogonal complement*?

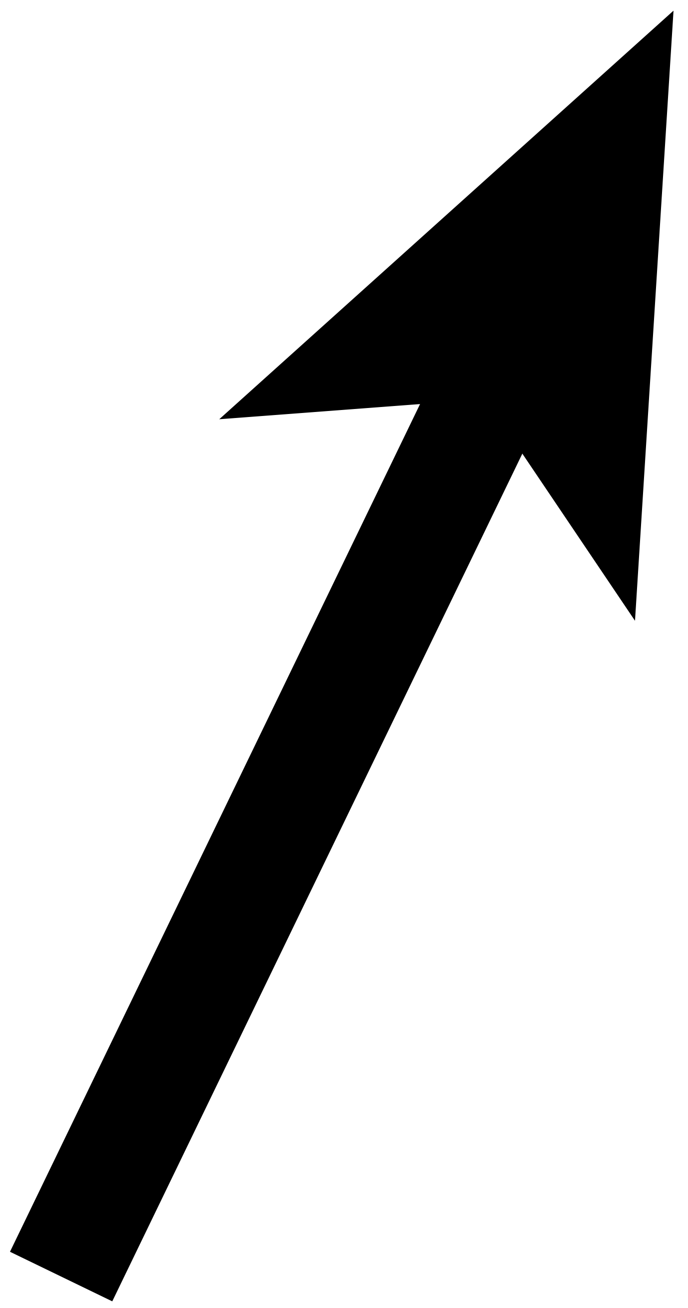


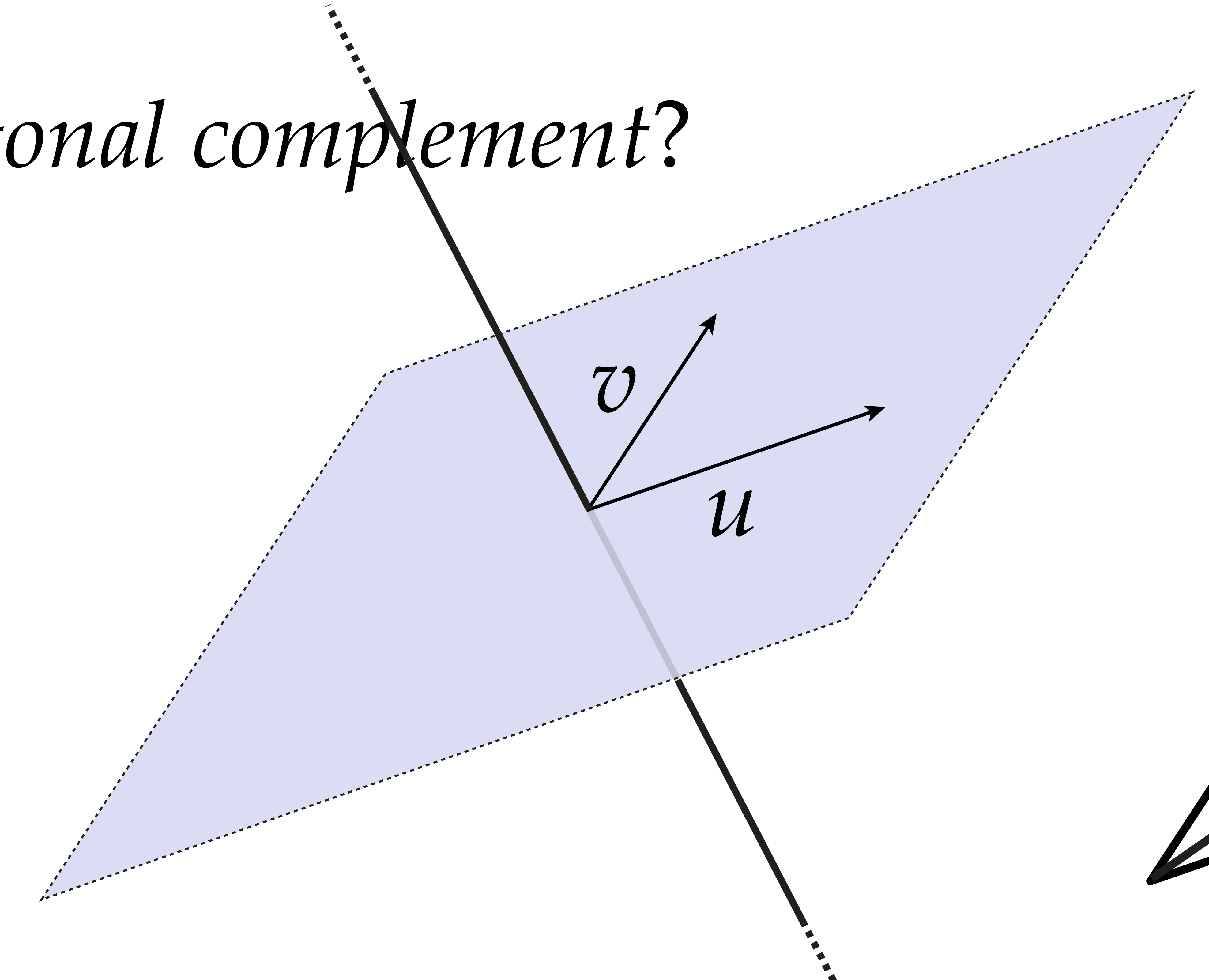


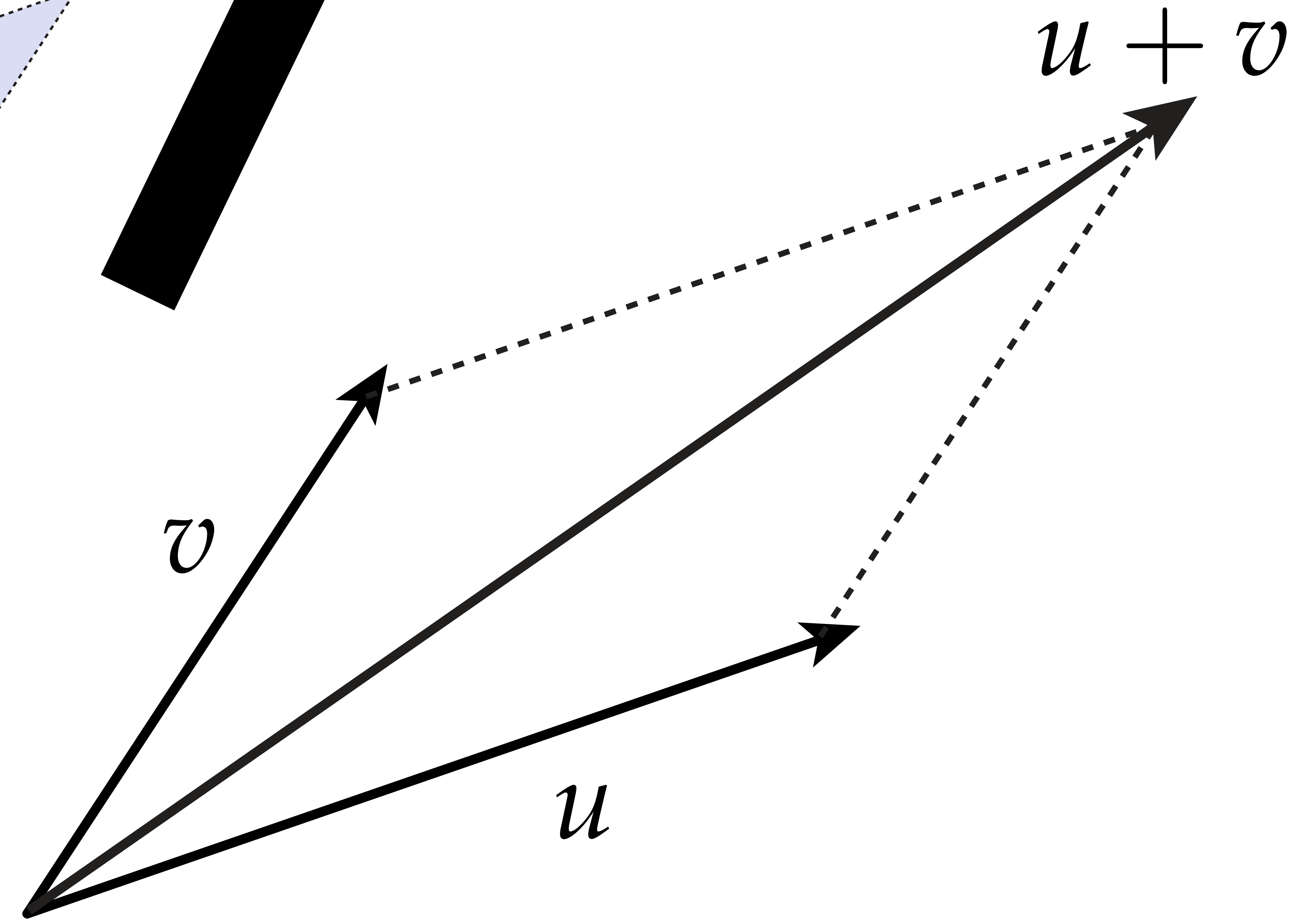


Review: Vector Spaces

- What is a vector? *(Geometrically?)*
- What is a vector *space*?
- What is the *span*?
- What is an *orthogonal complement*?





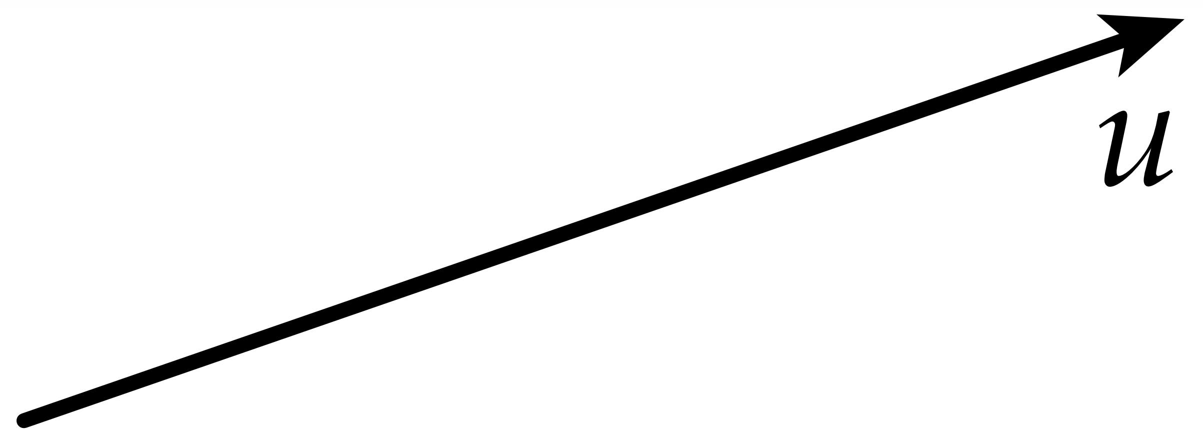


Wedge Product $(\wedge)$

Wedge Product $(\wedge)$

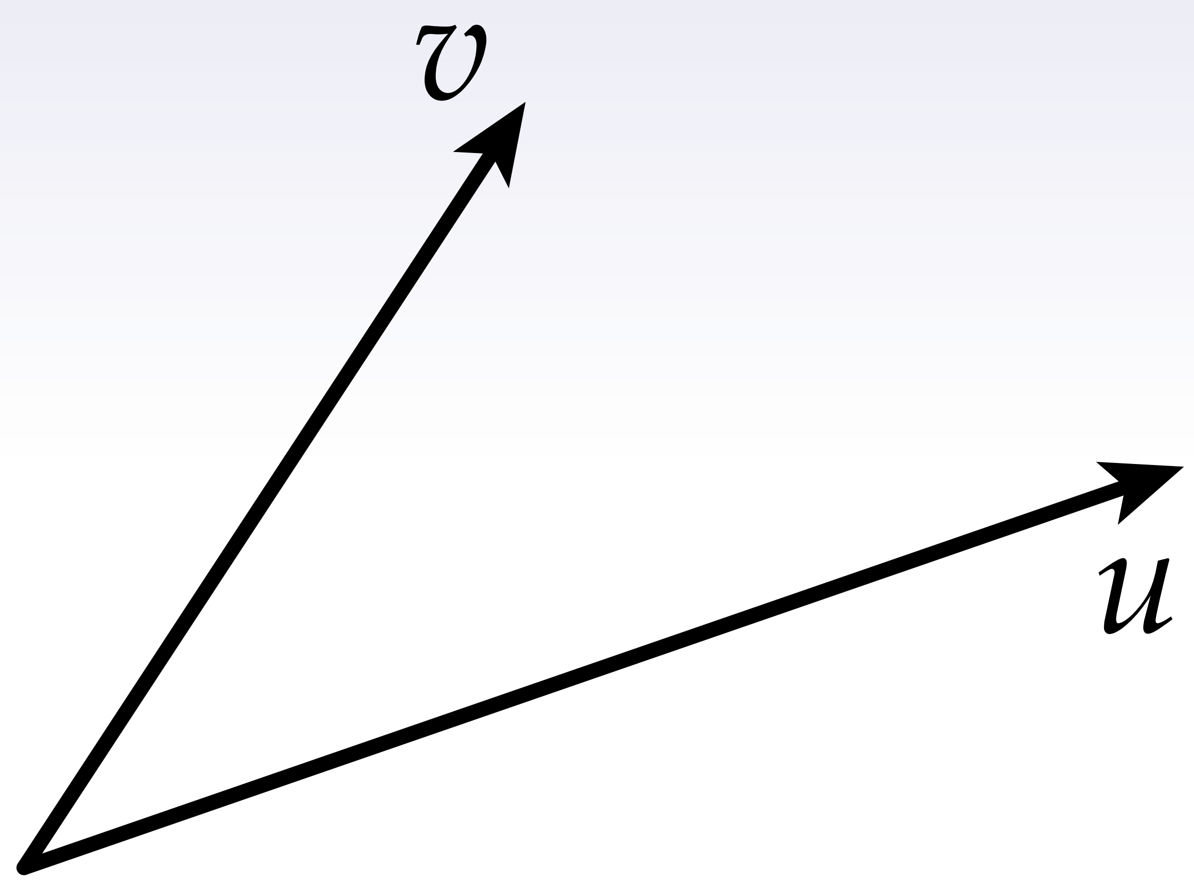
Analogy: *span*

Wedge Product $(\wedge)$



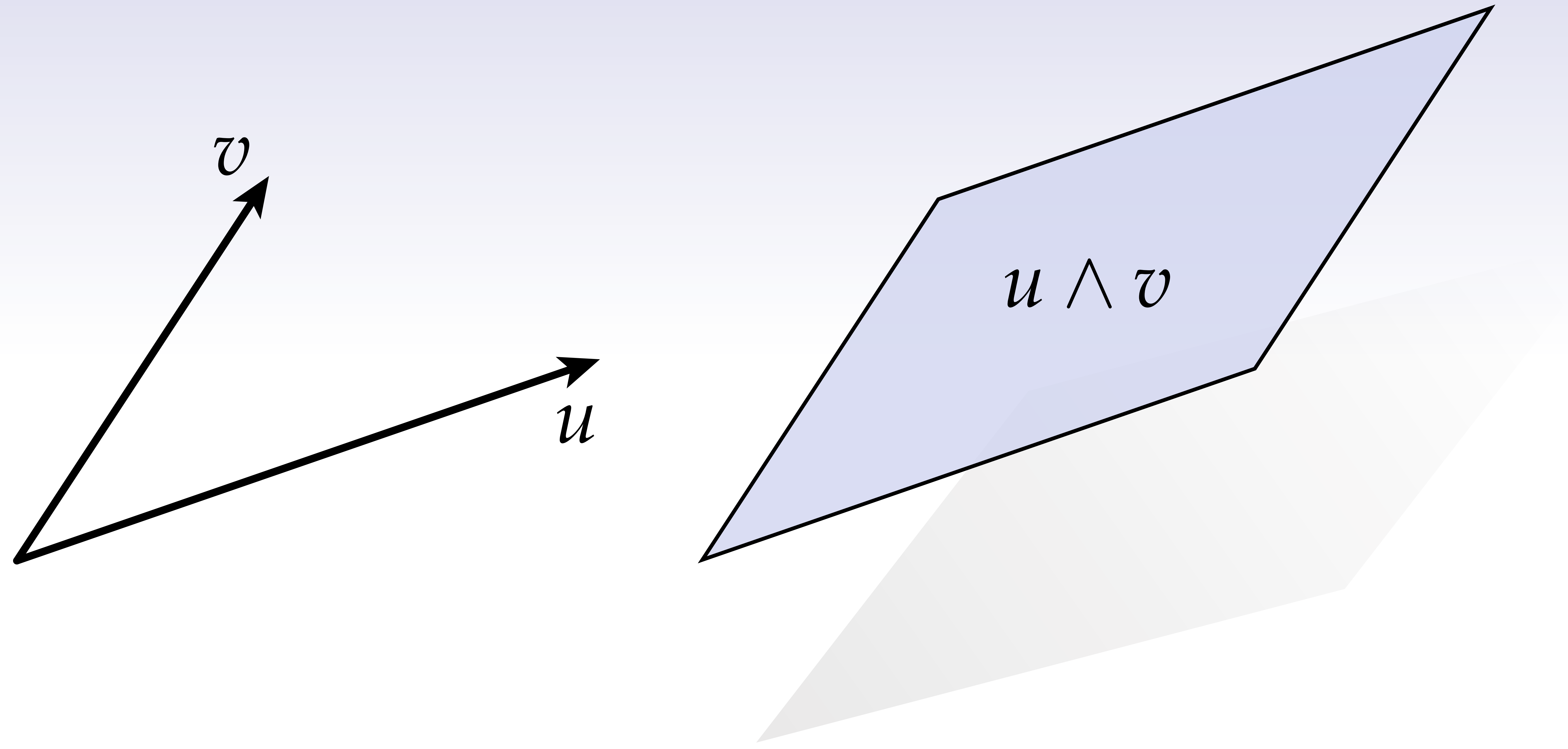
Analogy: *span*

Wedge Product $(\wedge)$



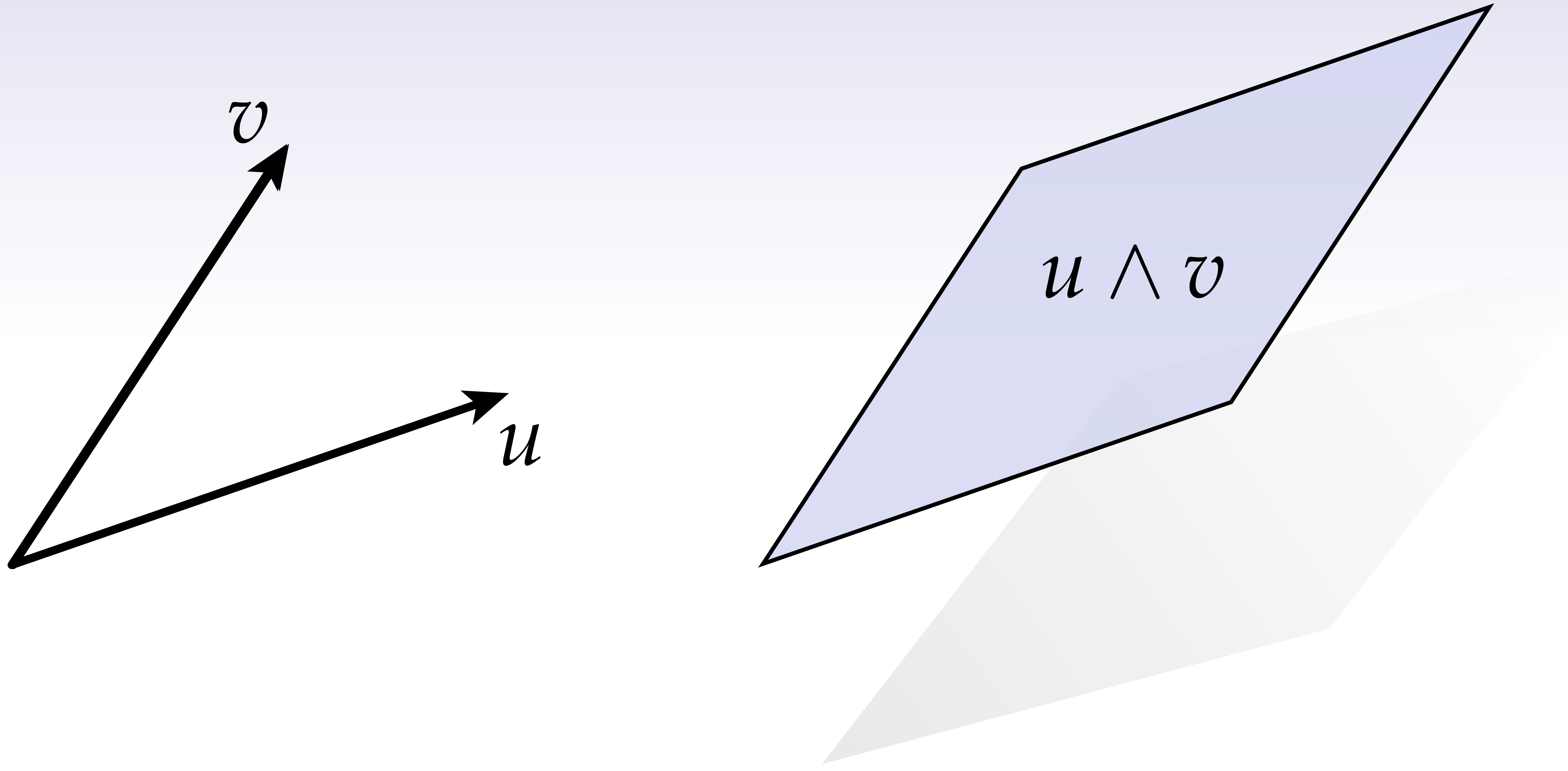
Analogy: *span*

Wedge Product $(\wedge)$



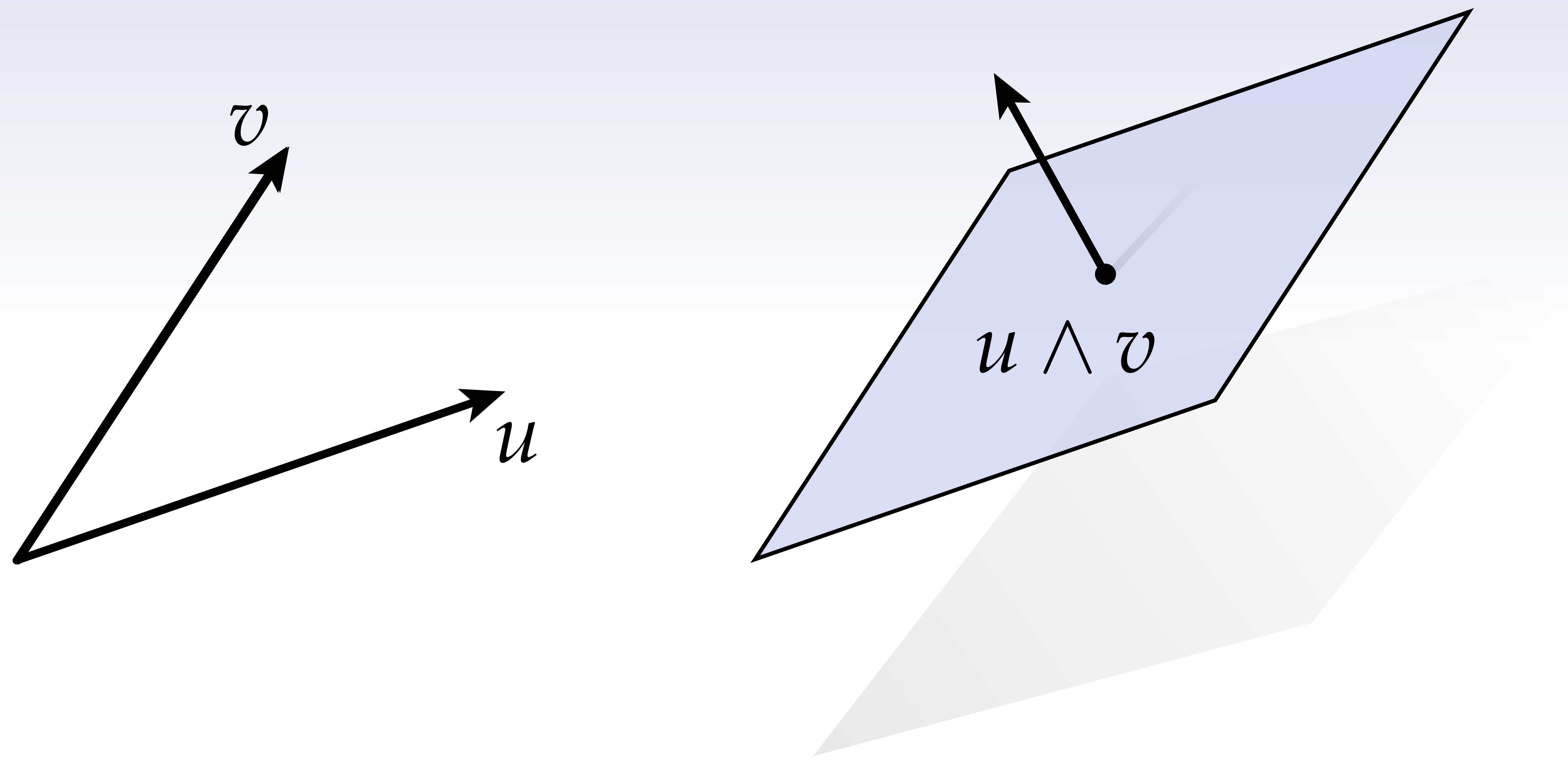
Analogy: *span*

Wedge Product $(\wedge)$



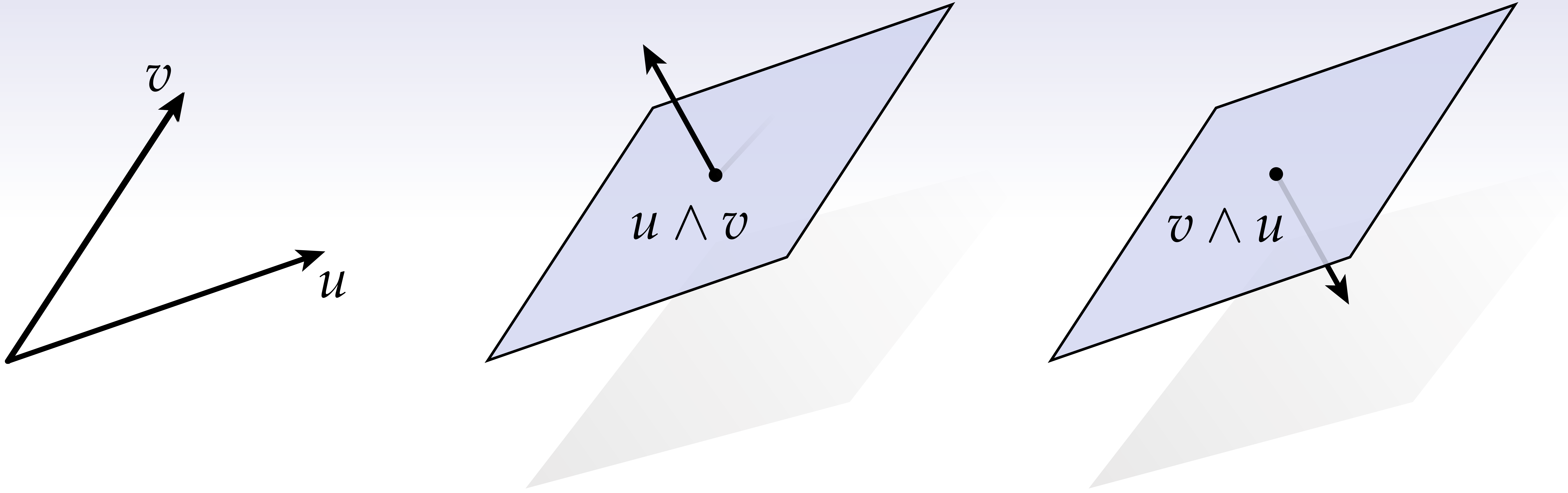
Analogy: *span*

Wedge Product $(\wedge)$



Analogy: *span*

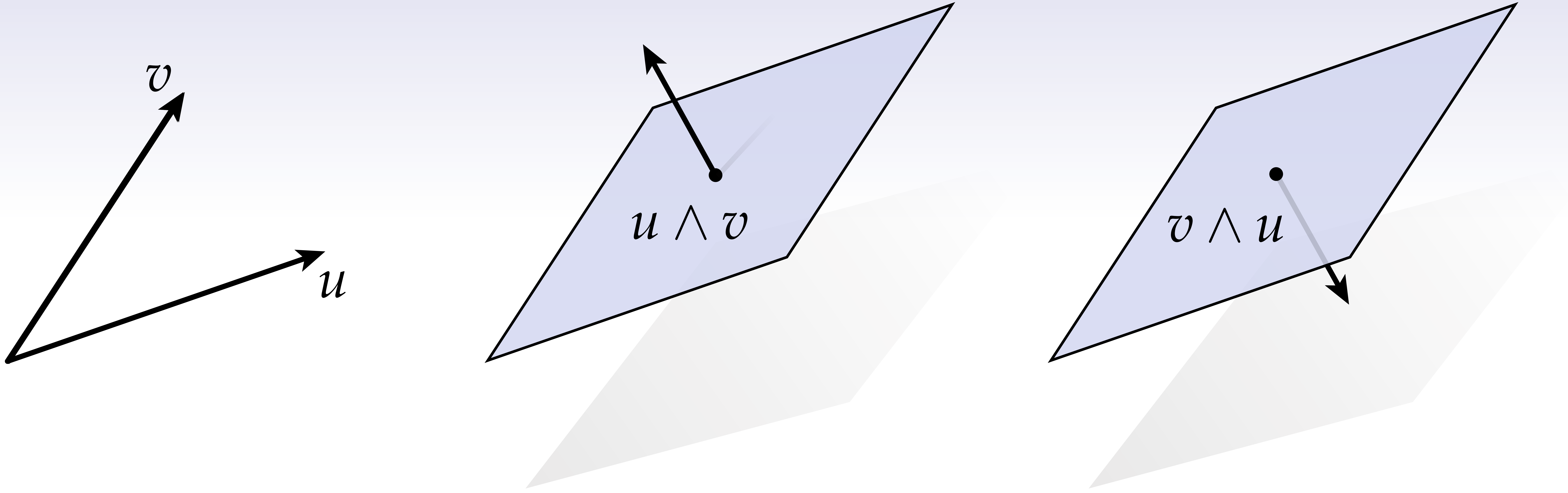
Wedge Product $(\wedge)$



Analogy: *span*

Wedge Product $(\wedge)$

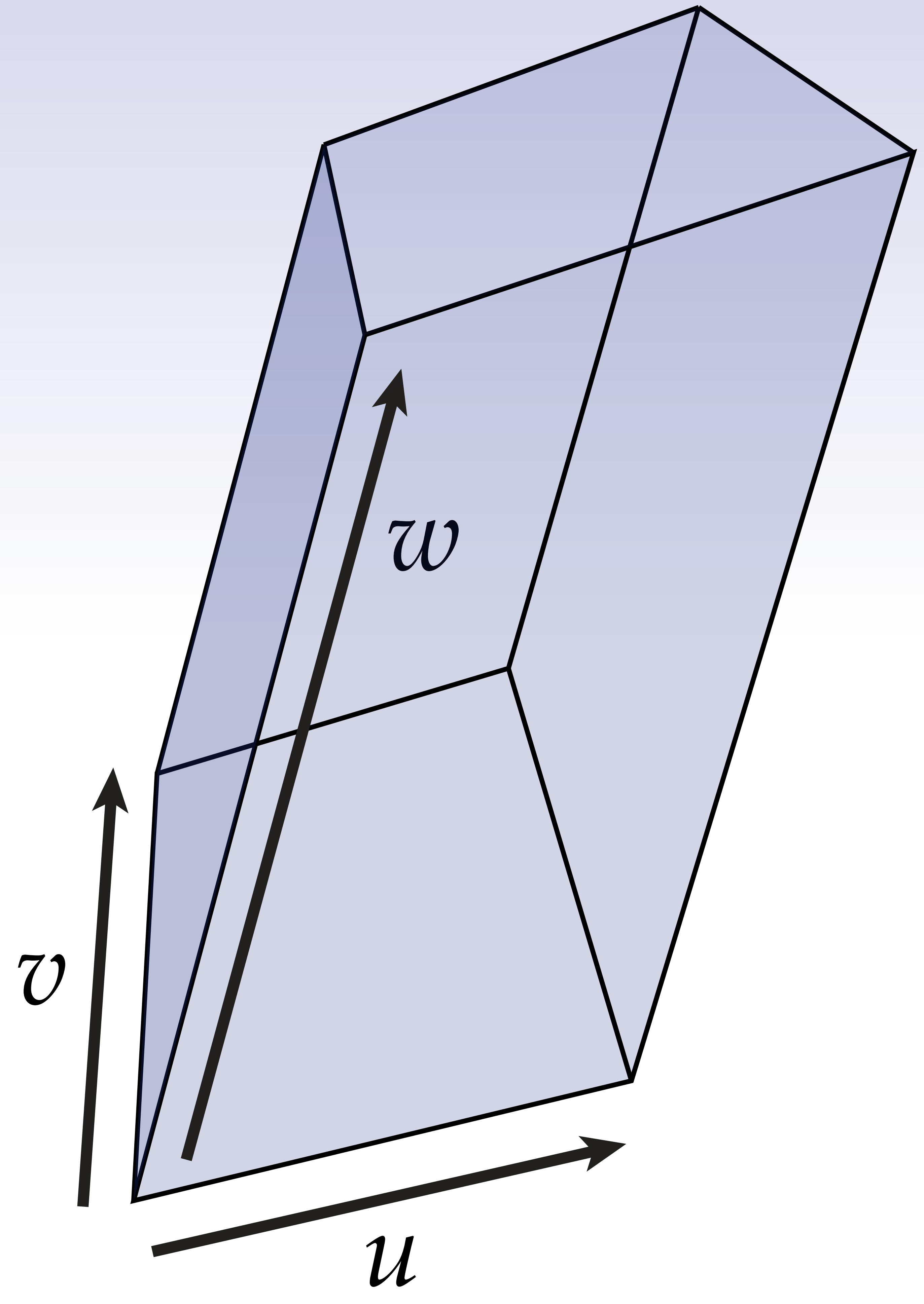
$$u \wedge v = -v \wedge u$$



Analogy: *span*

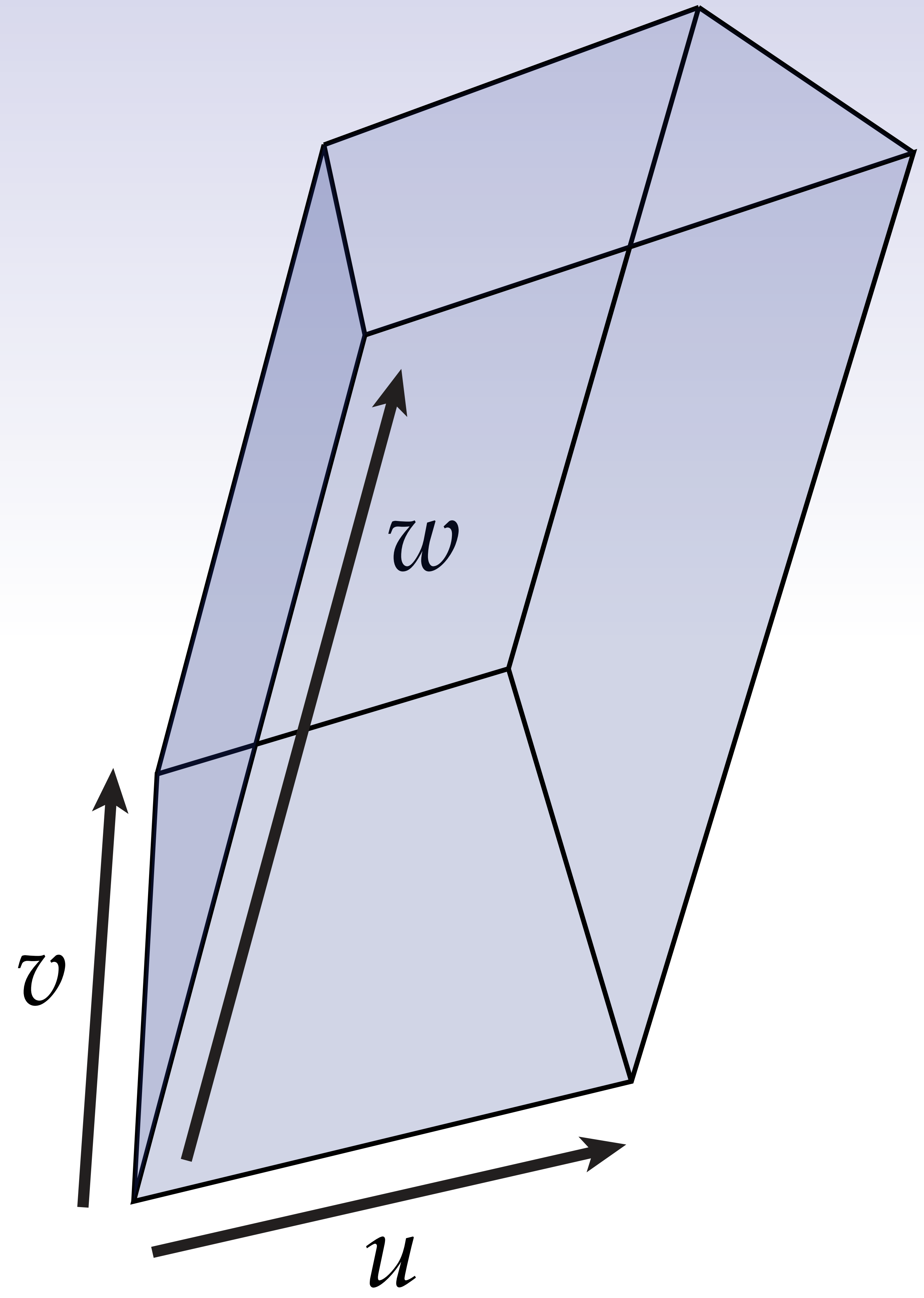
Wedge Product - Associativity

Wedge Product - Associativity

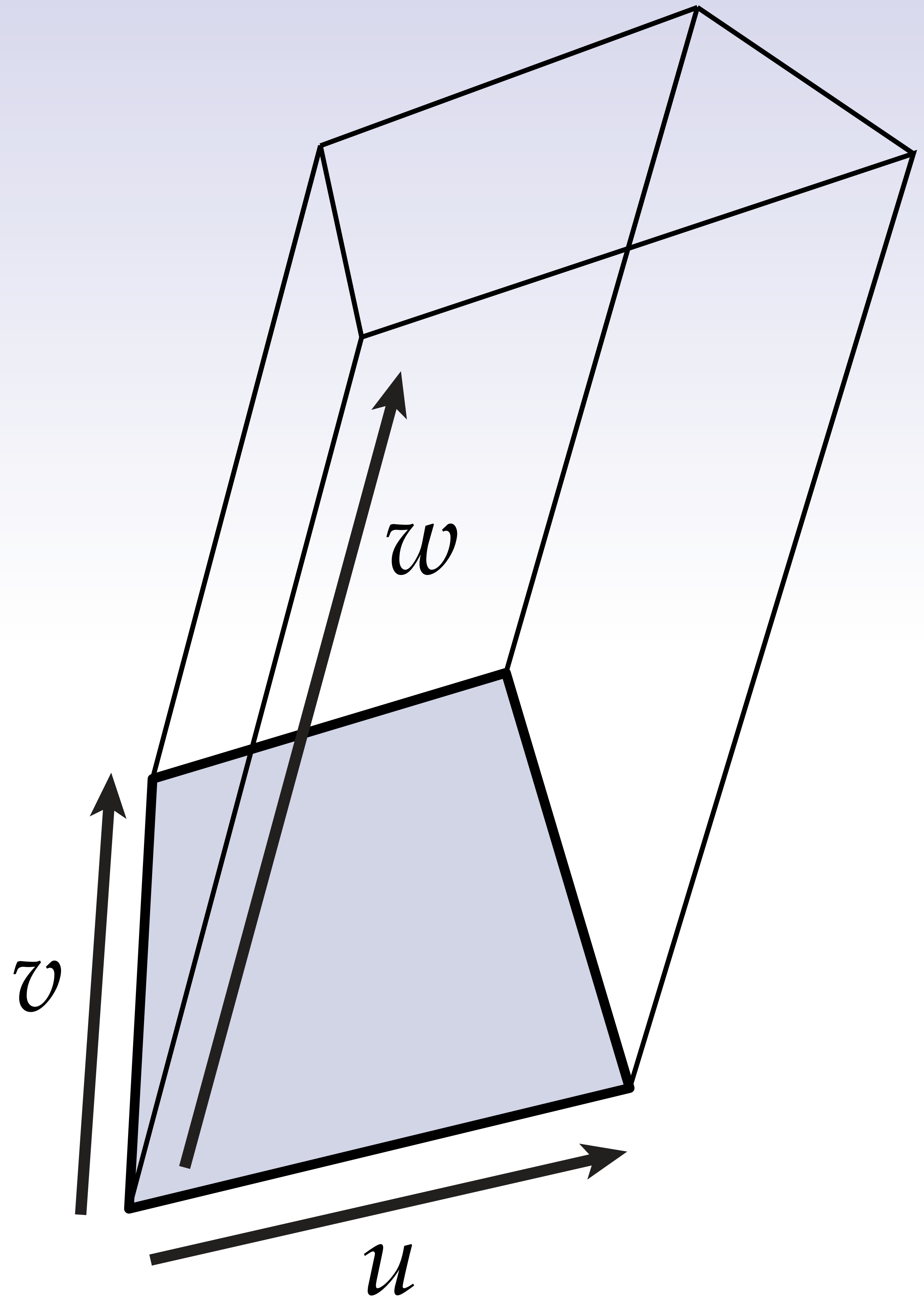


$$u \wedge v \wedge w$$

Wedge Product - Associativity

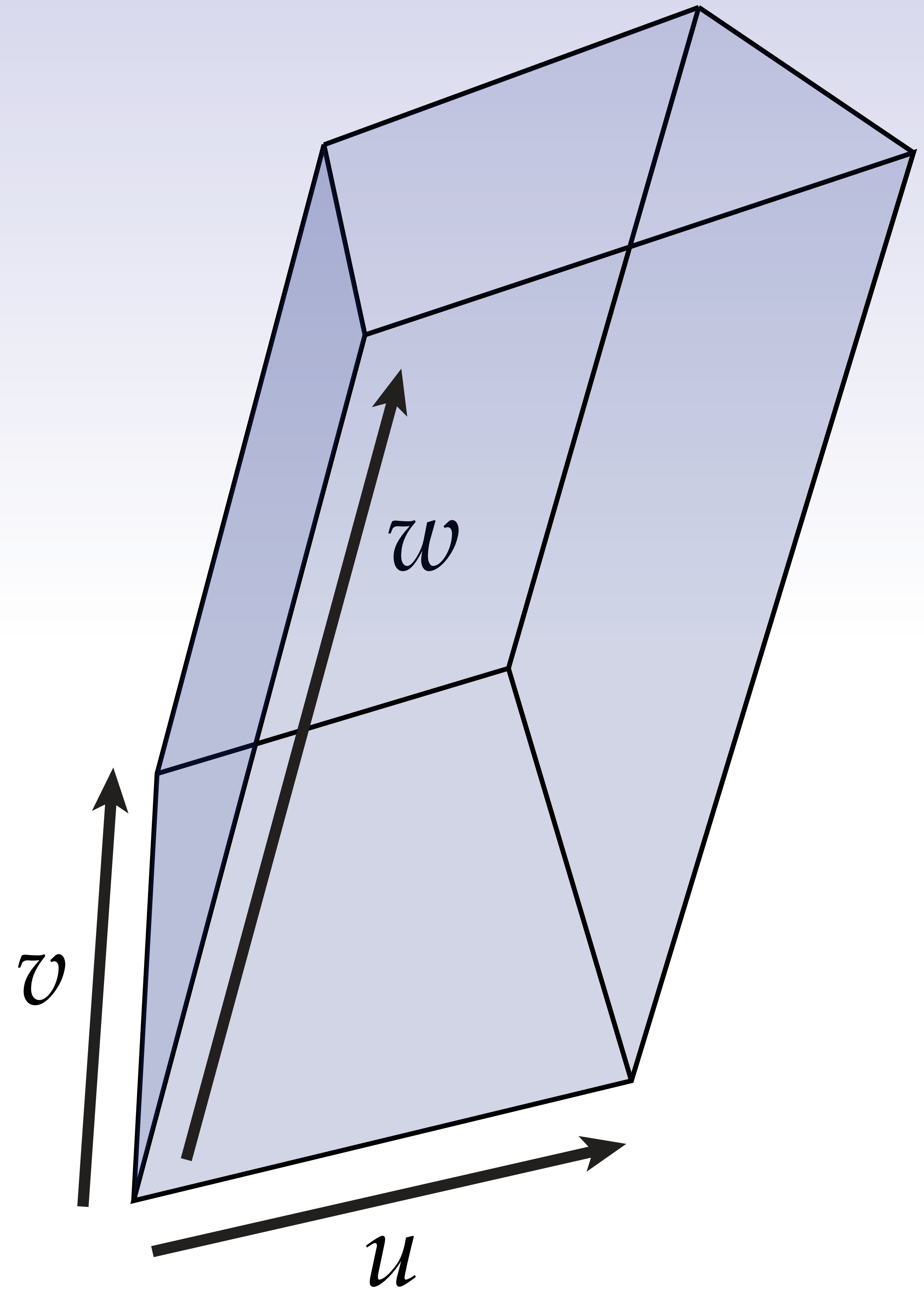


$u \wedge v \wedge w$

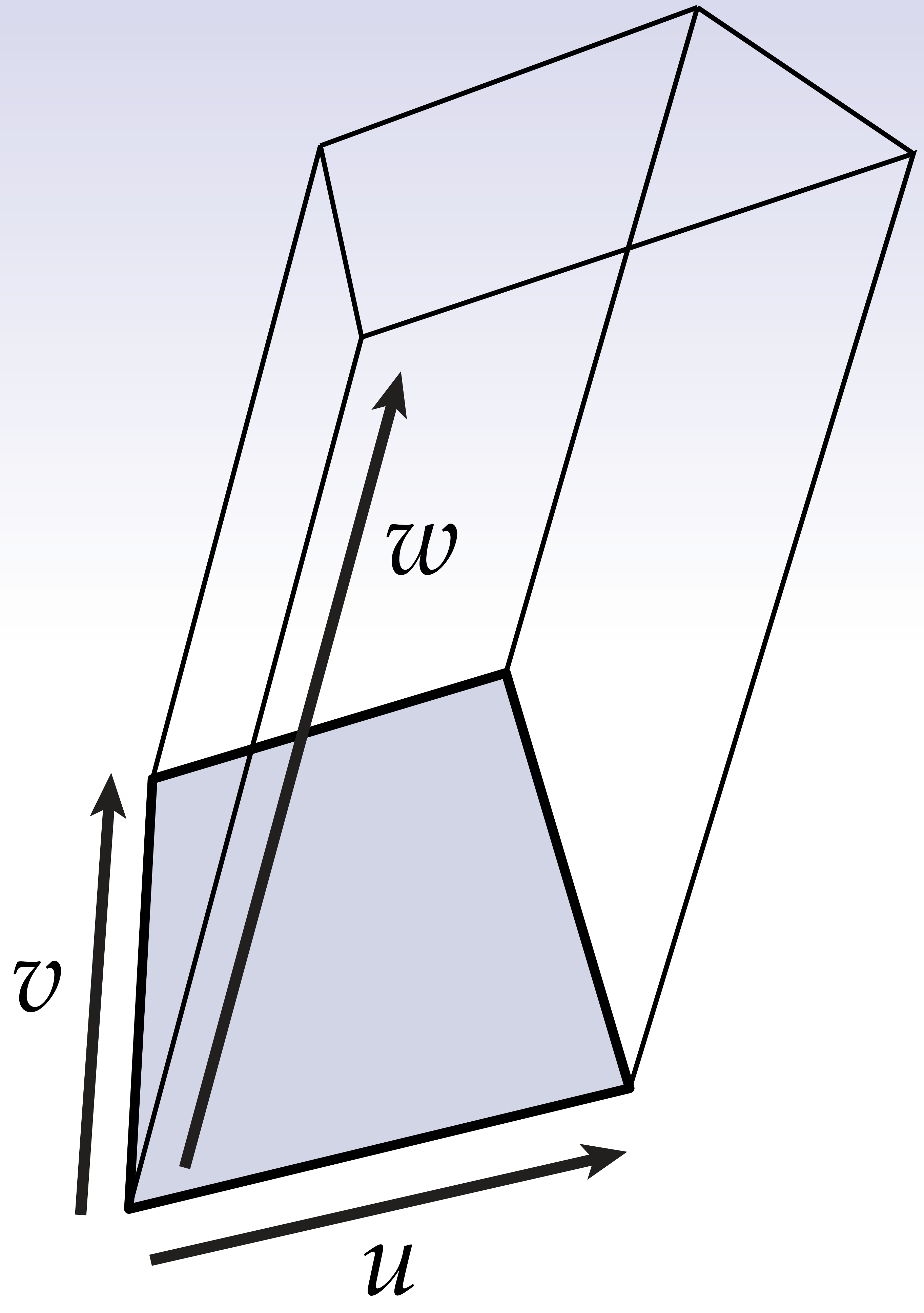


$(u \wedge v) \wedge w$

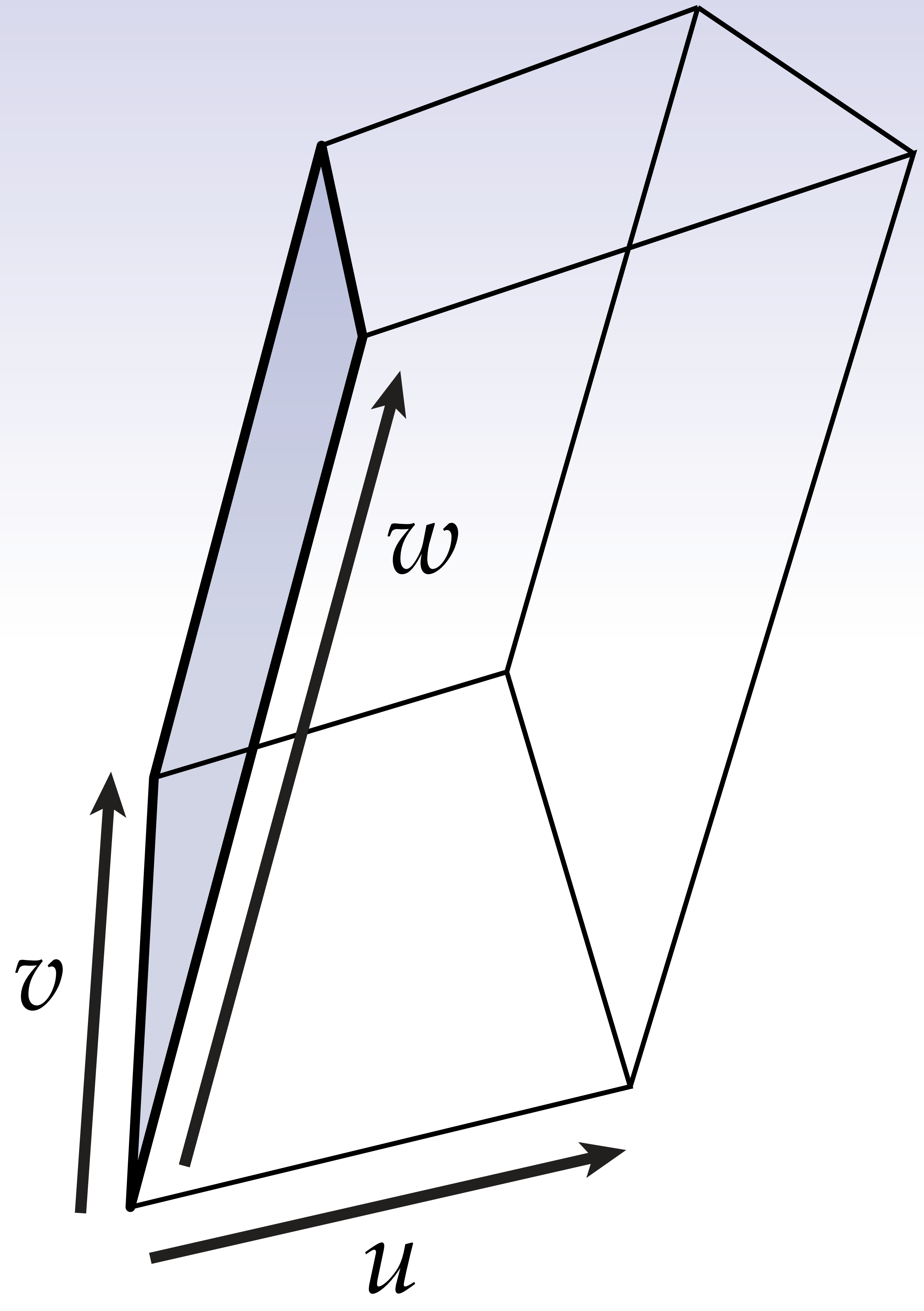
Wedge Product - Associativity



$u \wedge v \wedge w$



$(u \wedge v) \wedge w$

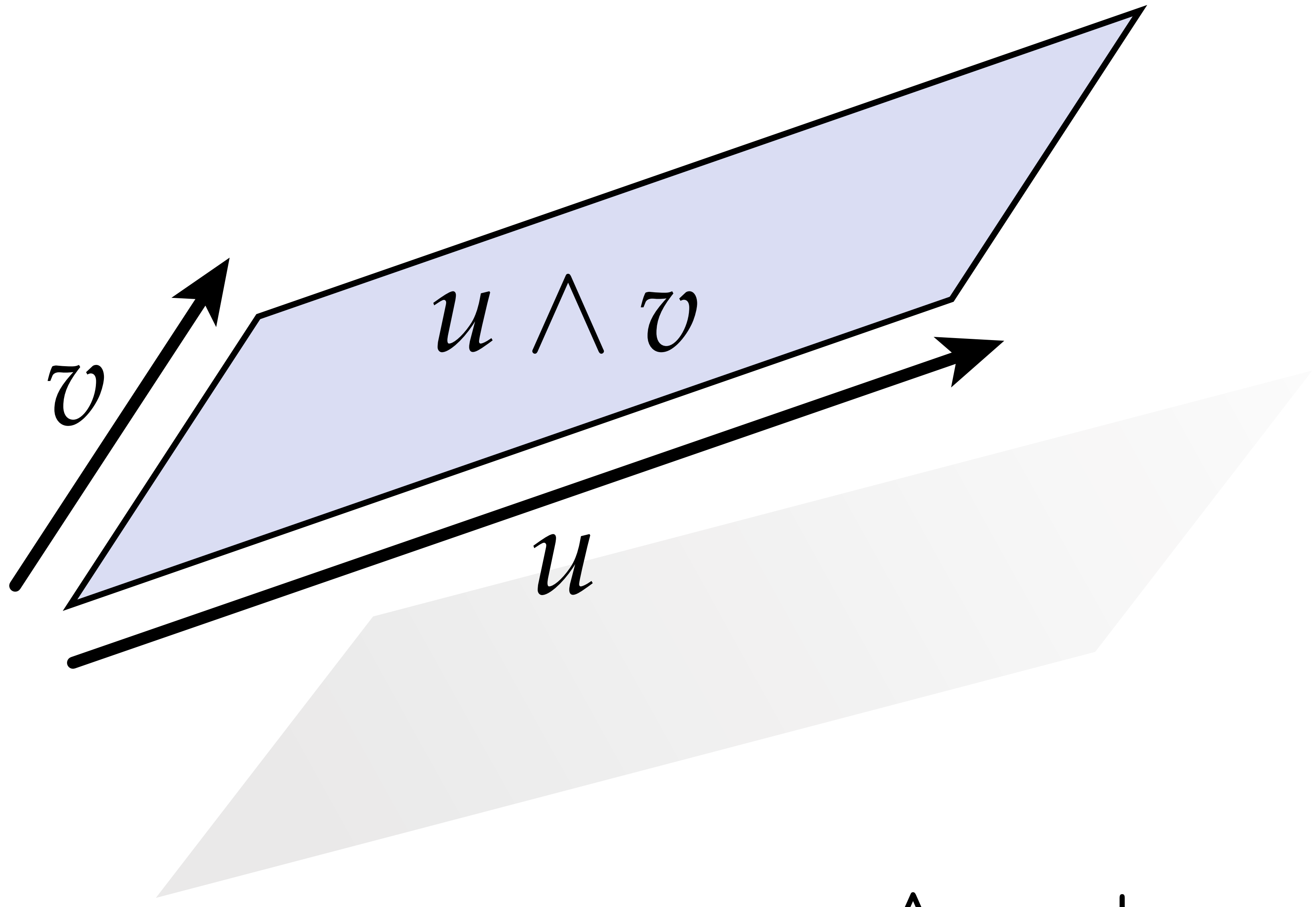


$u \wedge (v \wedge w)$

Wedge Product - Distributivity

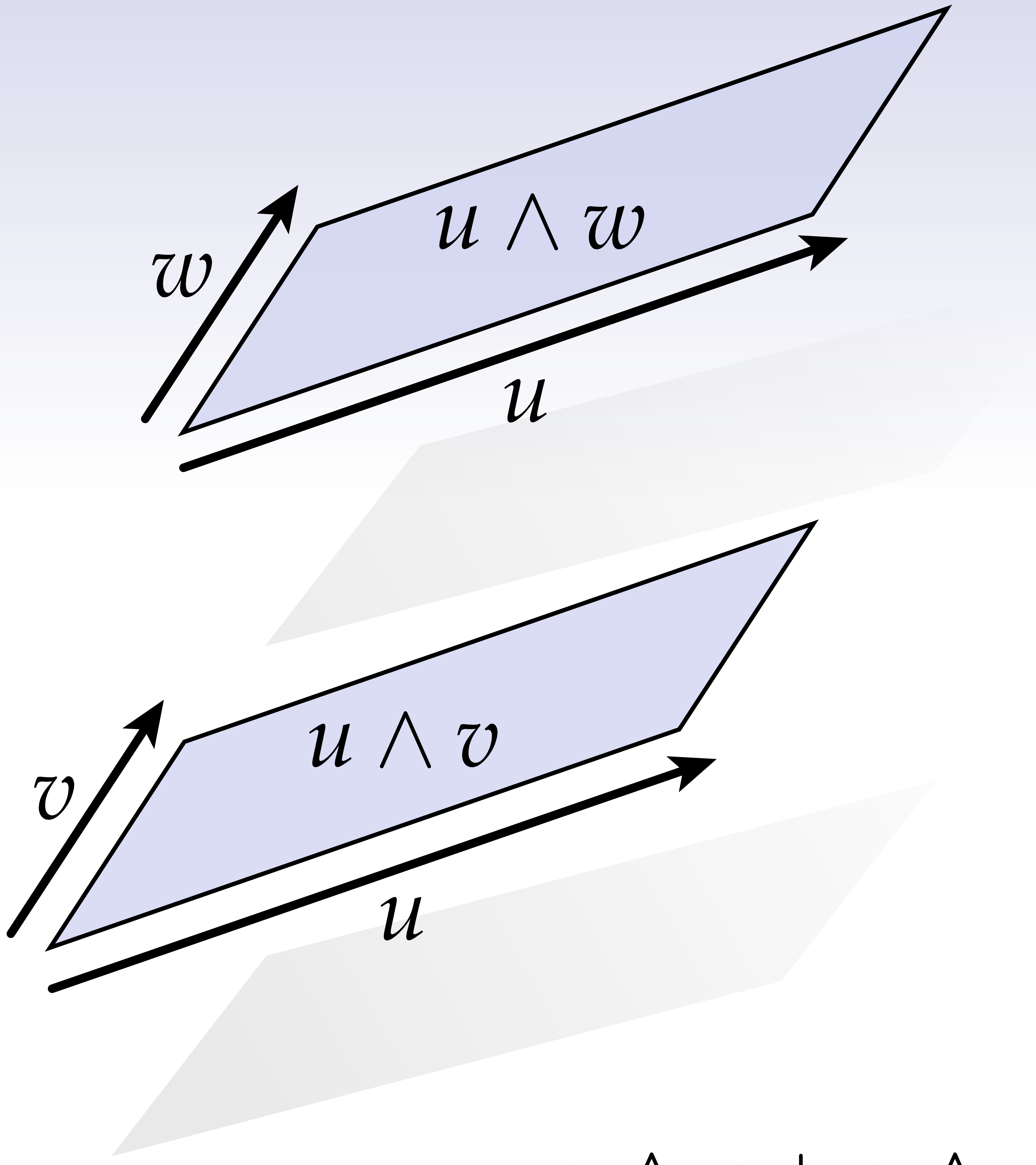
$$u \wedge v + u \wedge w = u \wedge (v + w)$$

Wedge Product - Distributivity



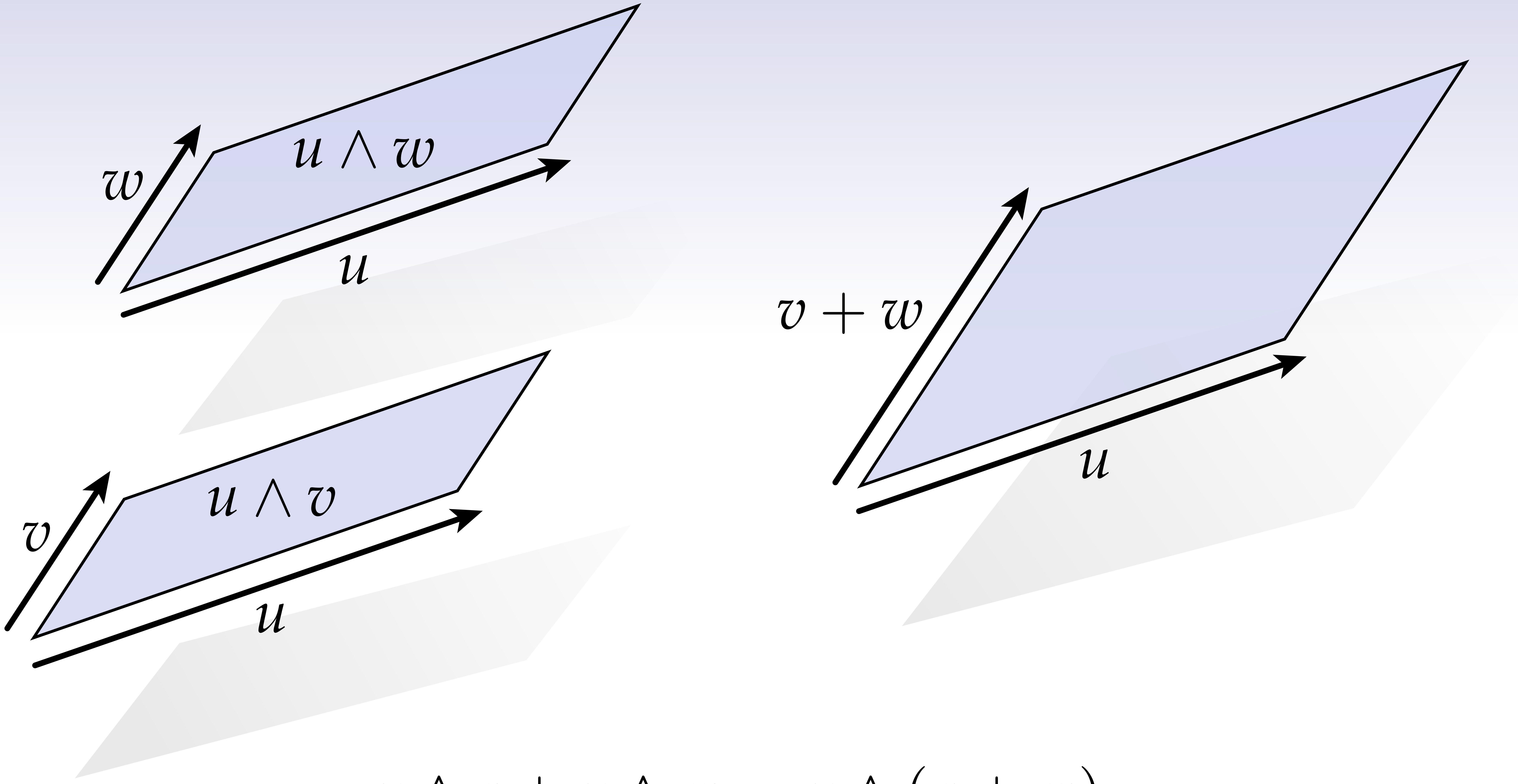
$$u \wedge v + u \wedge w = u \wedge (v + w)$$

Wedge Product - Distributivity



$$u \wedge v + u \wedge w = u \wedge (v + w)$$

Wedge Product - Distributivity



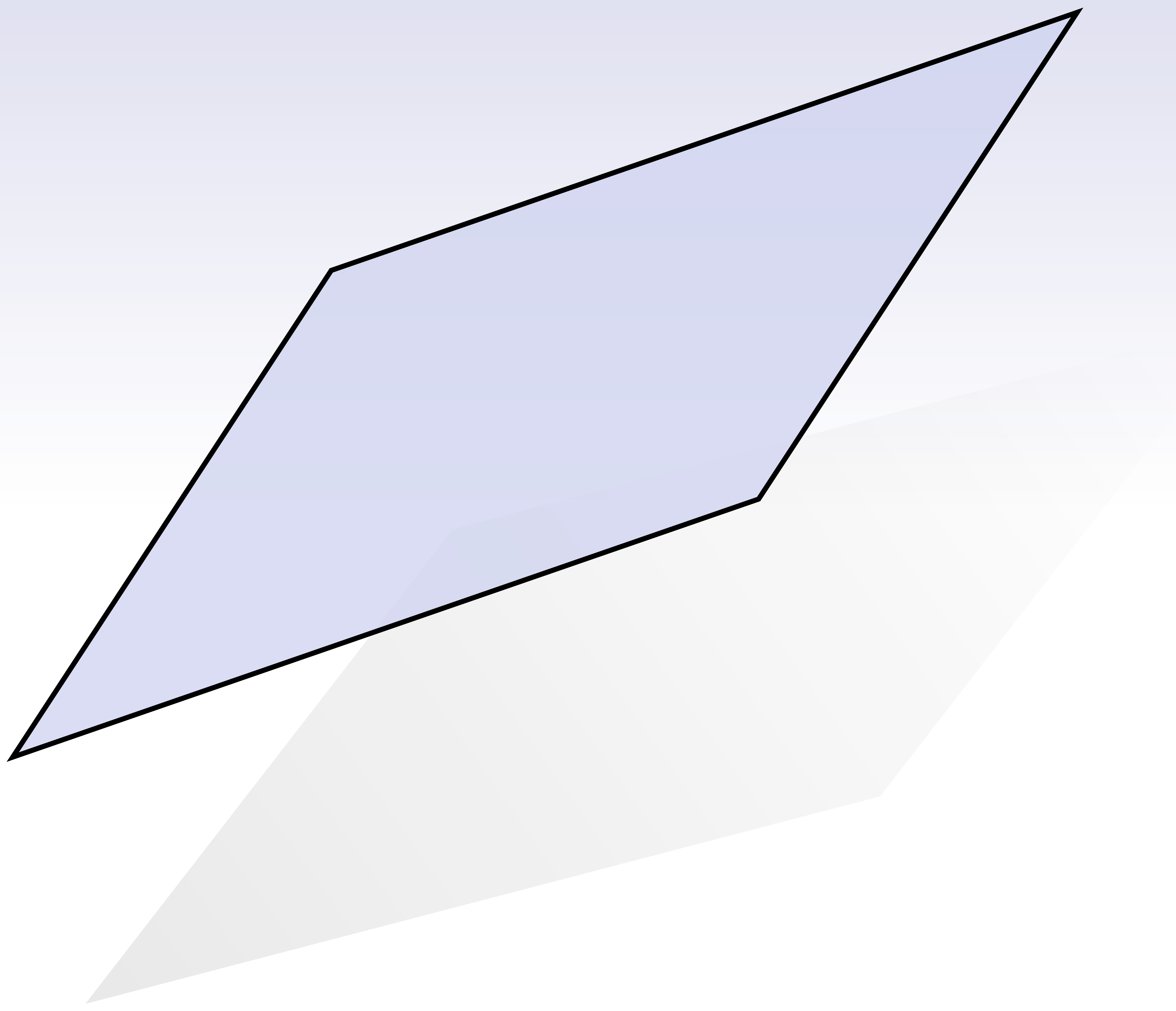
$$u \wedge v + u \wedge w = u \wedge (v + w)$$

Hodge Star $(\star)$

Hodge Star $(\star)$

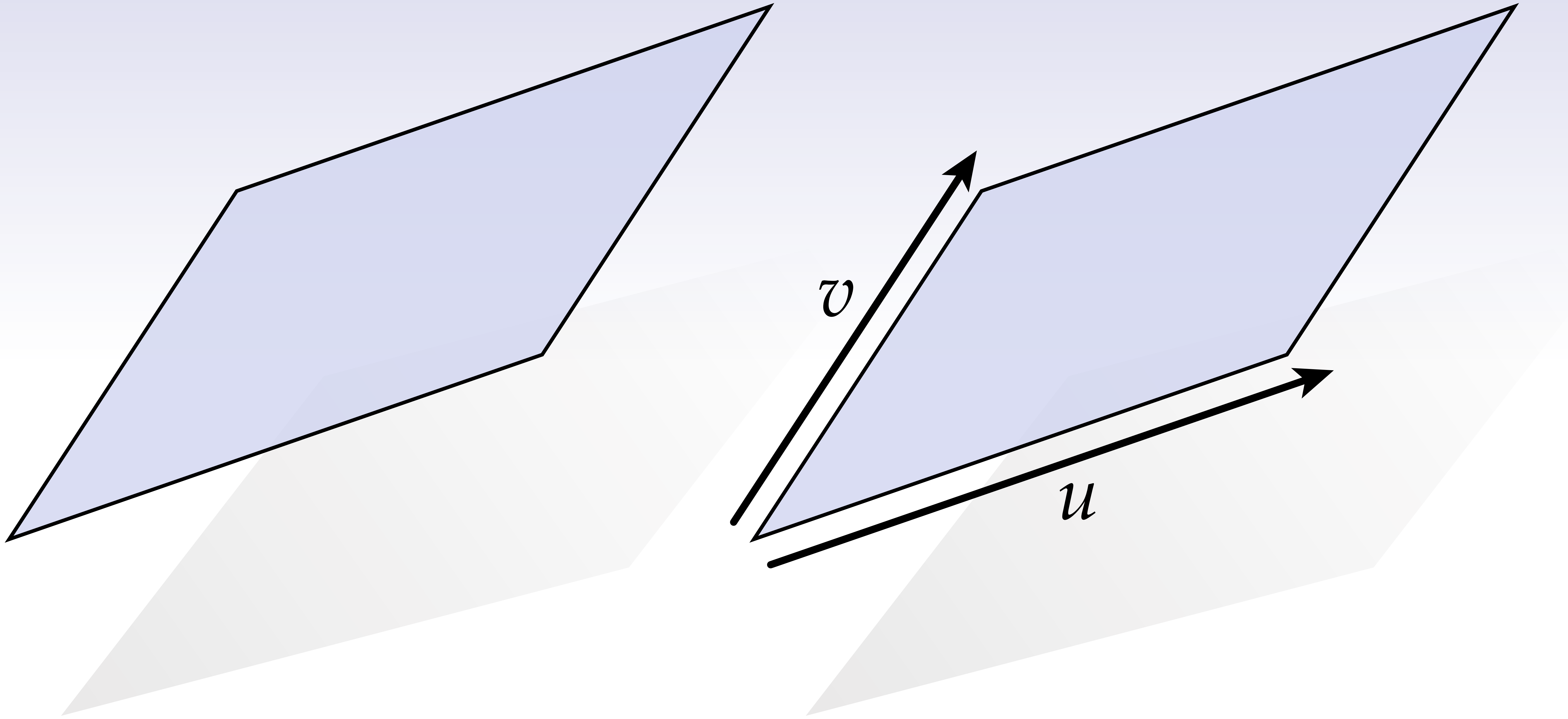
Analogy: *orthogonal complement*

Hodge Star $(\star)$



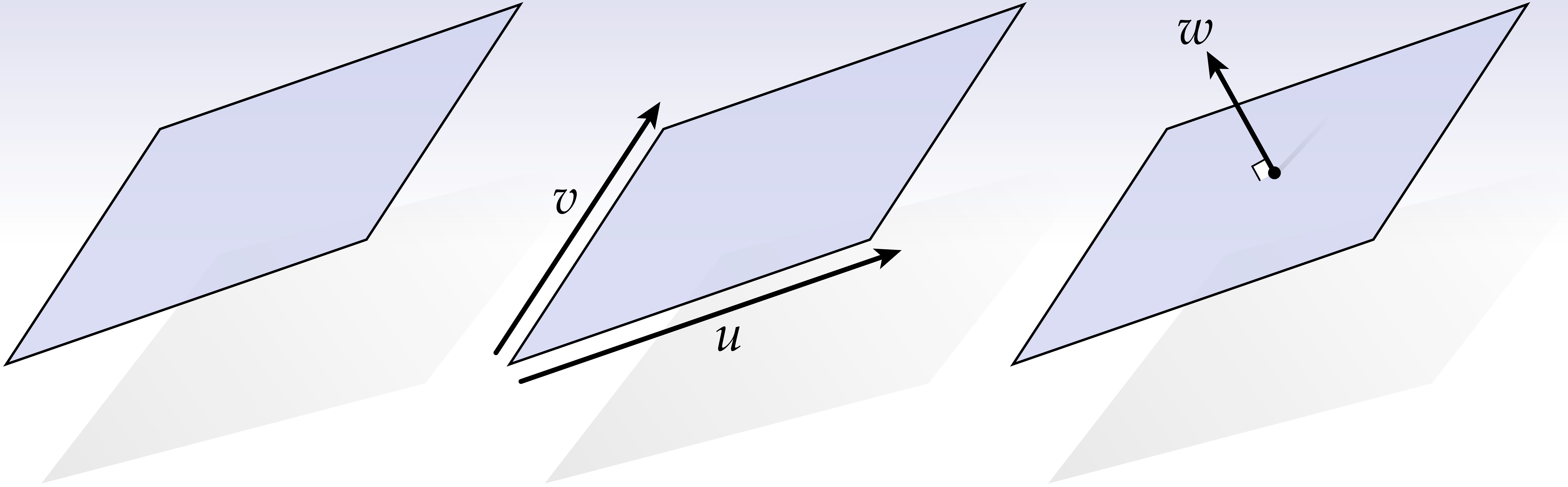
Analogy: *orthogonal complement*

Hodge Star $(\star)$



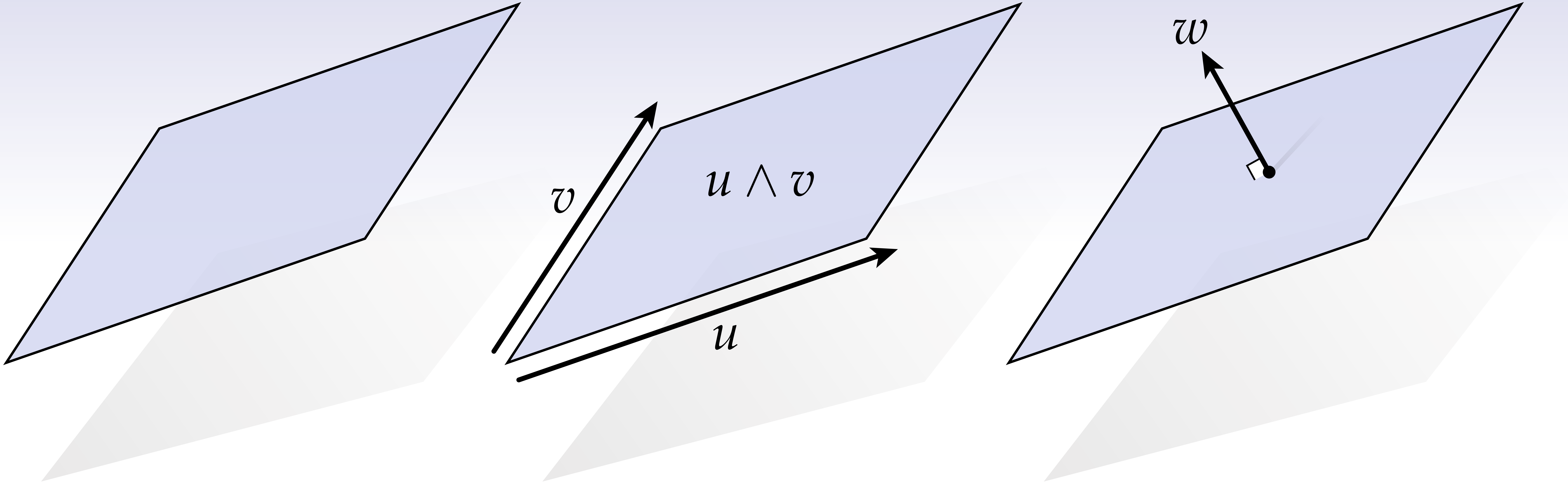
Analogy: *orthogonal complement*

Hodge Star $(\star)$



Analogy: *orthogonal complement*

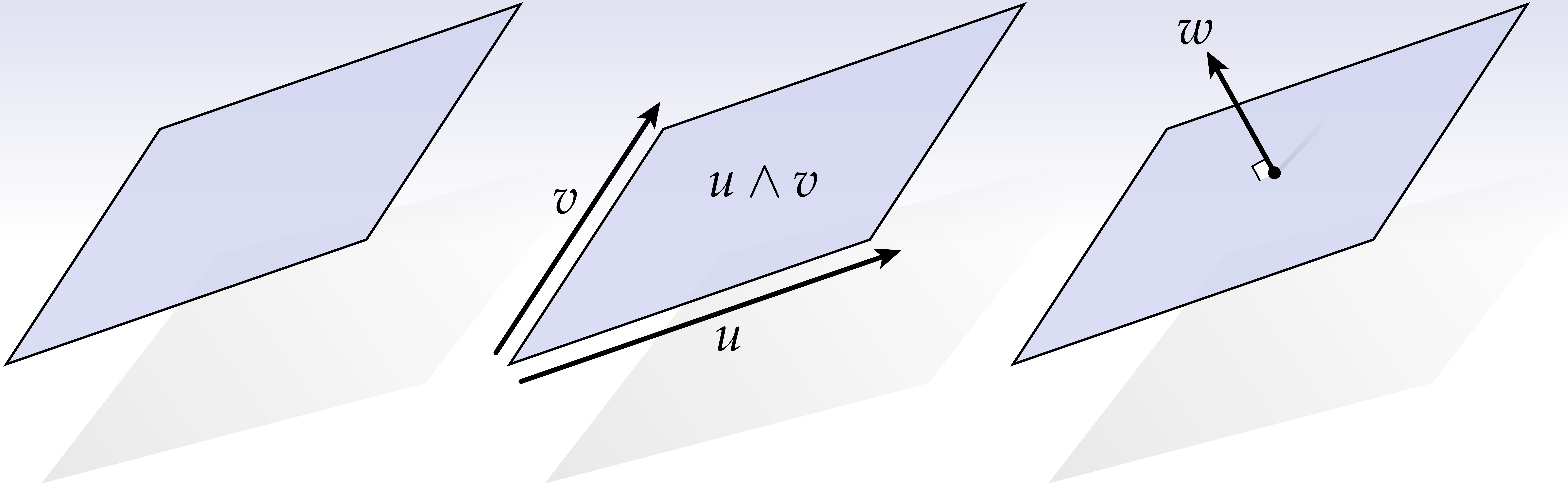
Hodge Star $(\star)$



Analogy: *orthogonal complement*

Hodge Star ($\star$)

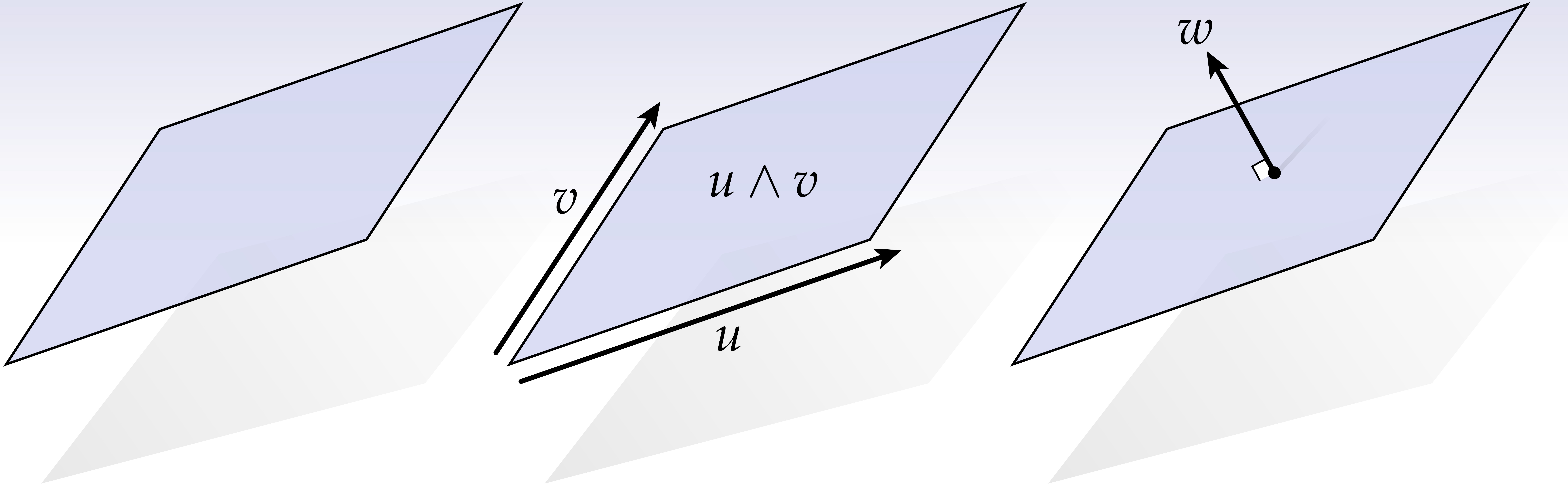
$\star u \wedge v = w$



Analogy: *orthogonal complement*

Hodge Star $(\star)$

$\star u \wedge v = w$



Analogy: *orthogonal complement*

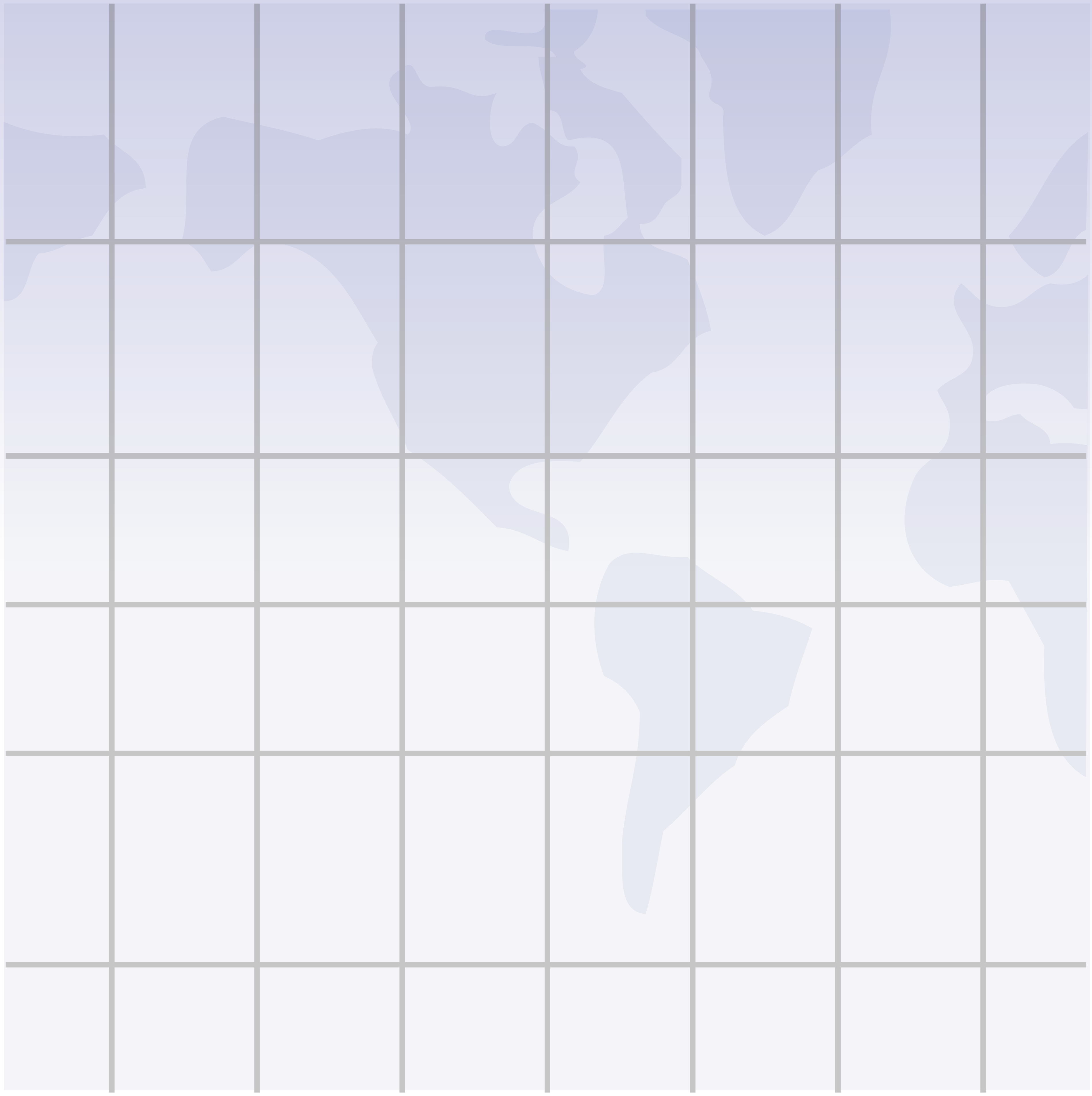
$k \mapsto (n-k)$

Hodge Star - 2D

Hodge Star - 2D

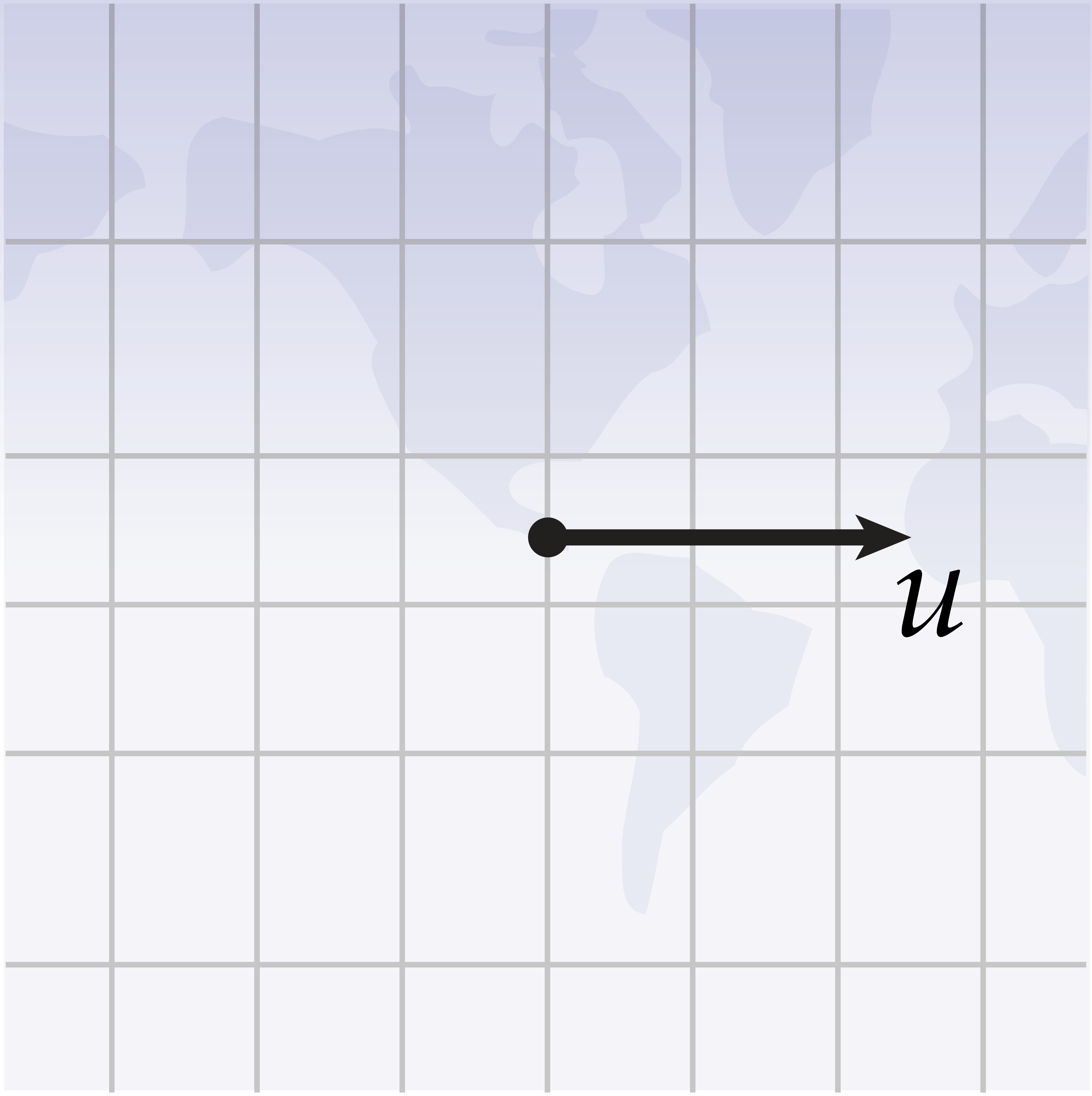
Analogy: *90-degree rotation*

Hodge Star - 2D



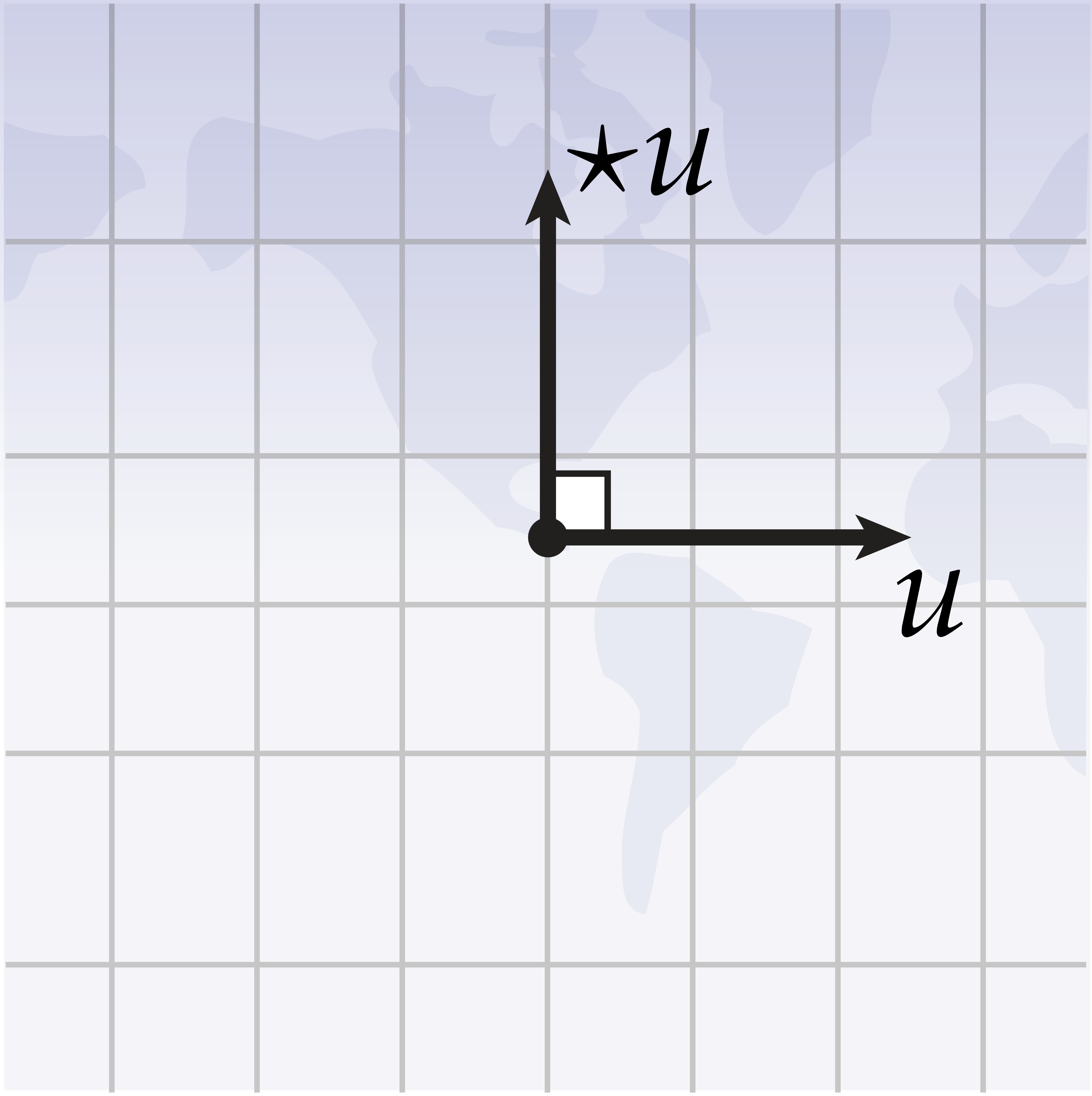
Analogy: *90-degree rotation*

Hodge Star - 2D



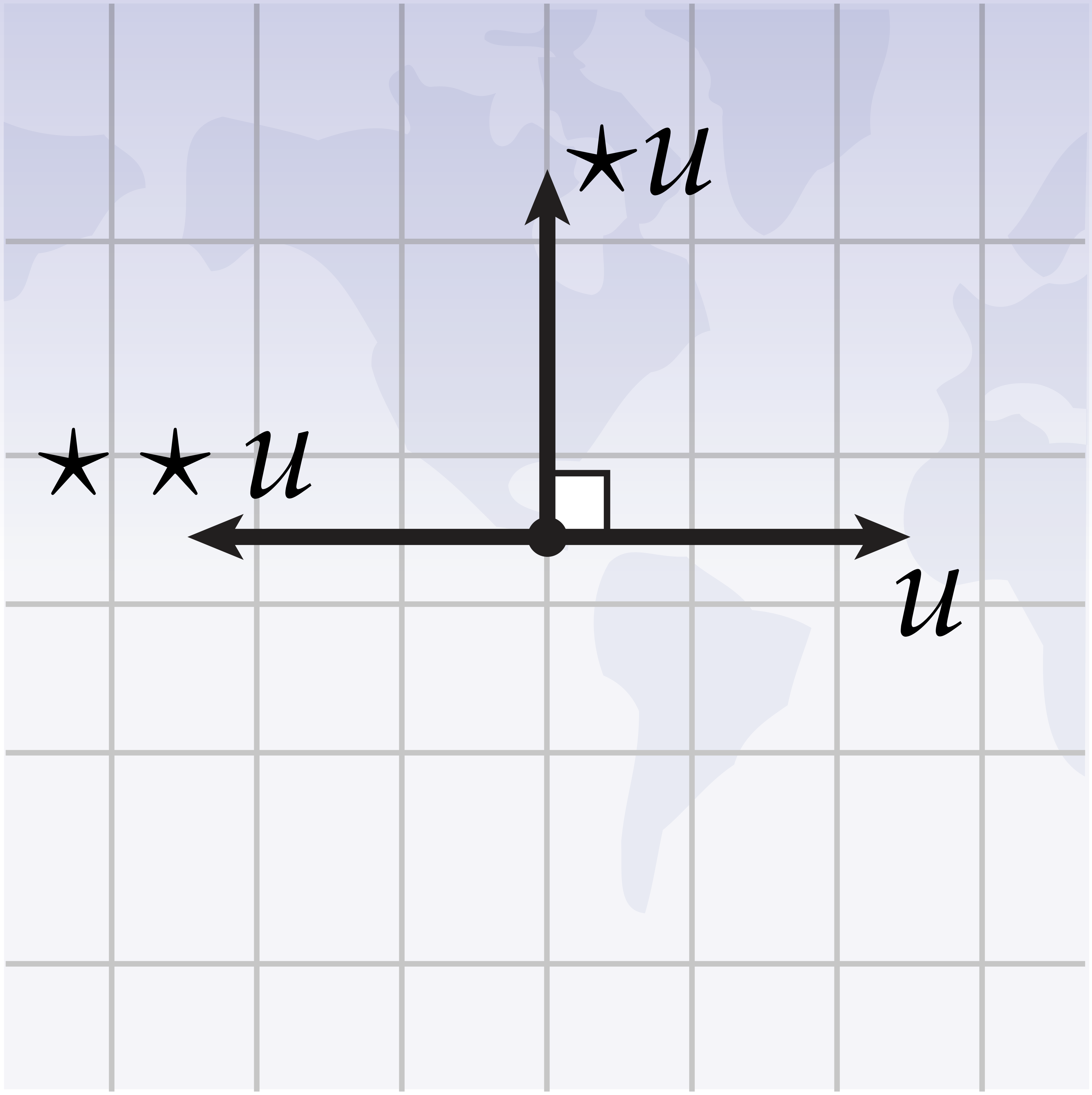
Analogy: *90-degree rotation*

Hodge Star - 2D



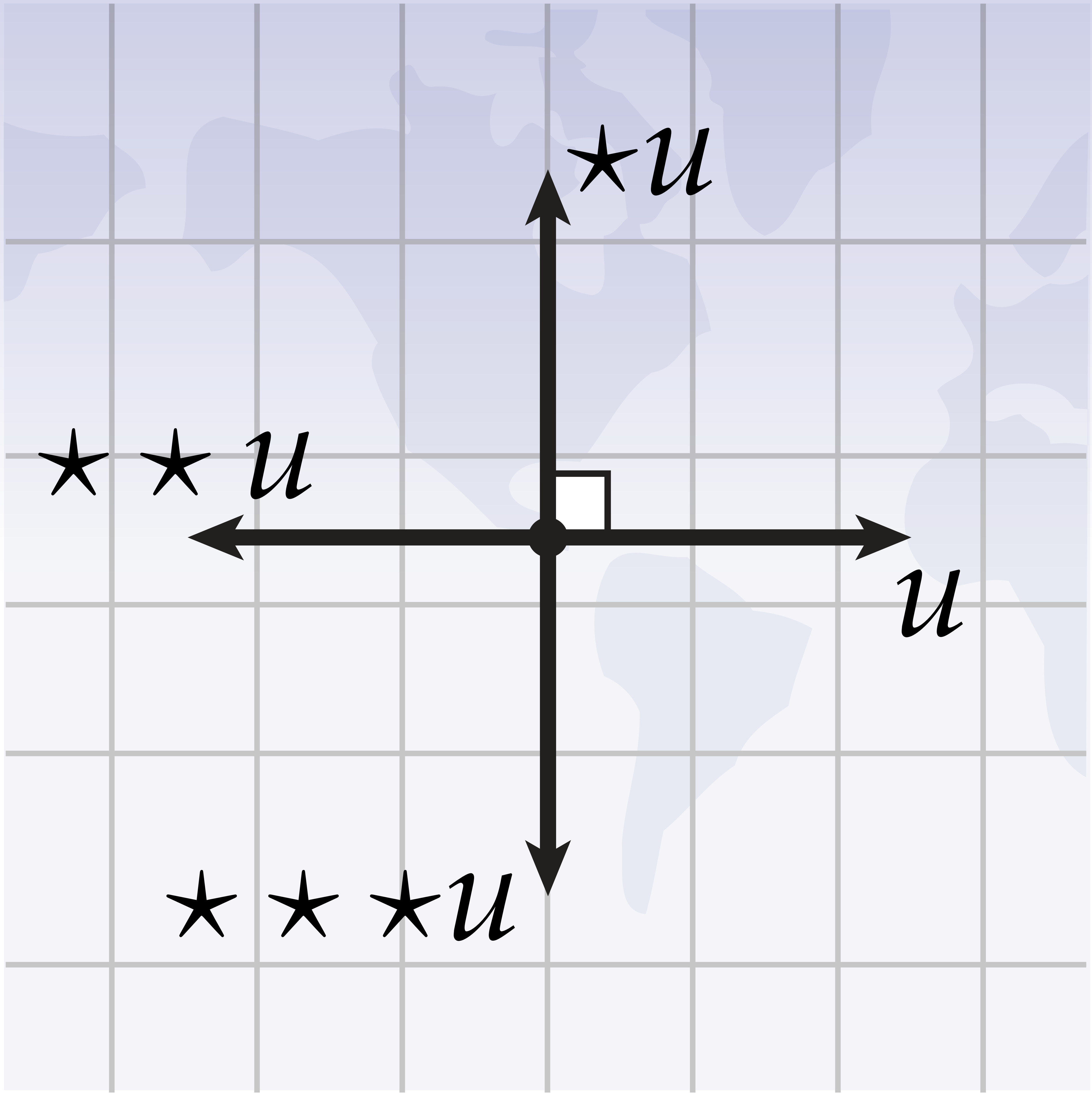
Analogy: *90-degree rotation*

Hodge Star - 2D



Analogy: *90-degree rotation*

Hodge Star - 2D



Analogy: *90-degree rotation*

Exterior Algebra - Summary

Exterior Algebra - Summary

- Wedge product

Exterior Algebra - Summary

- Wedge product
  - analogous to *span* of vectors

Exterior Algebra - Summary

- Wedge product
  - analogous to *span* of vectors
  - length matters (magnitude)

Exterior Algebra - Summary

- Wedge product
  - analogous to *span* of vectors
  - length matters (magnitude)
  - order matters (orientation)

Exterior Algebra - Summary

- Wedge product
  - analogous to *span* of vectors
  - length matters (magnitude)
  - order matters (orientation)
- Hodge star

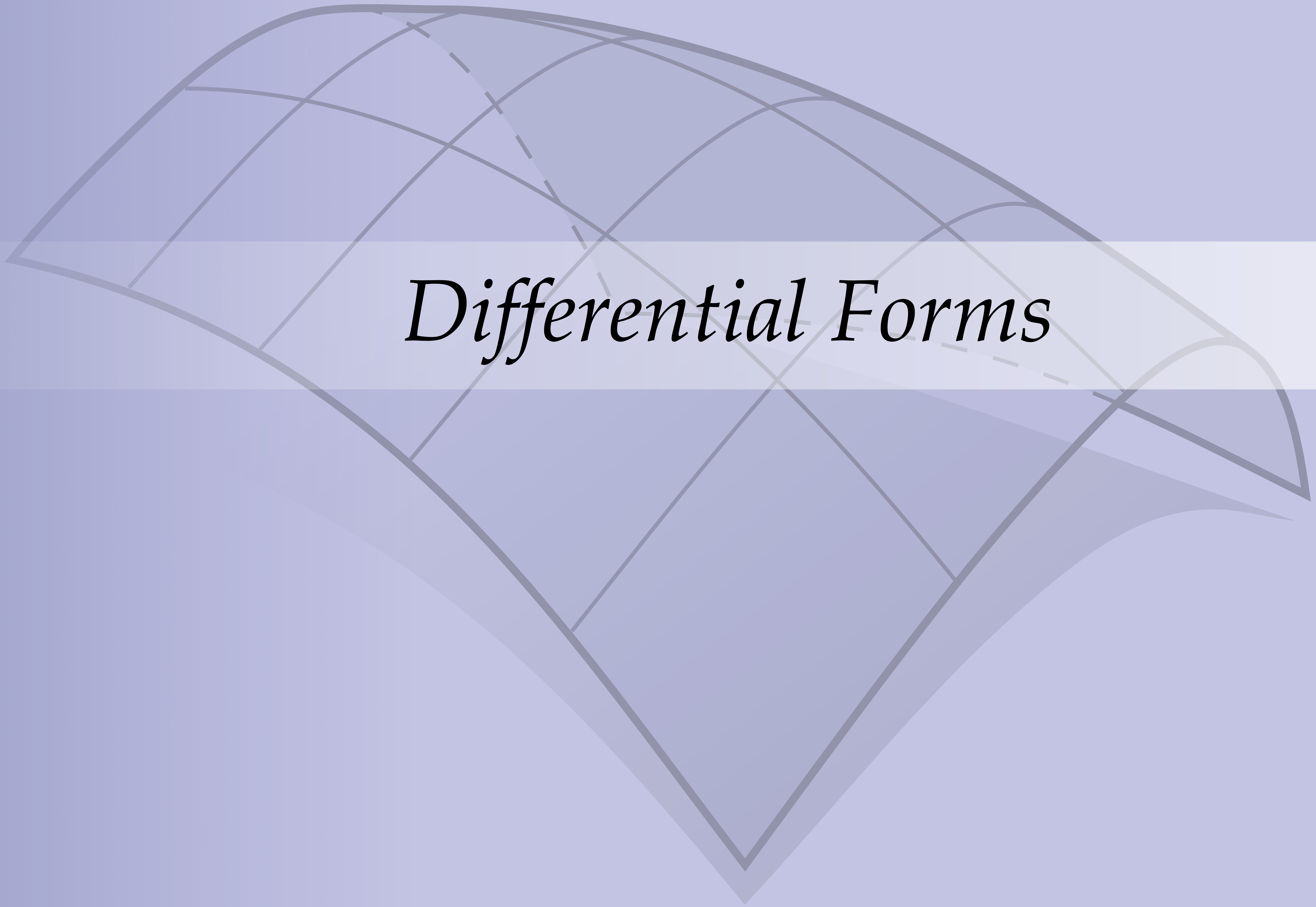
Exterior Algebra - Summary

- Wedge product
  - analogous to *span* of vectors
  - length matters (magnitude)
  - order matters (orientation)
- Hodge star
  - analogous to *orthogonal complement*

Exterior Algebra - Summary

- Wedge product
  - analogous to *span* of vectors
  - length matters (magnitude)
  - order matters (orientation)
- Hodge star
  - analogous to *orthogonal complement*
  - in 2D: 90-degree rotation

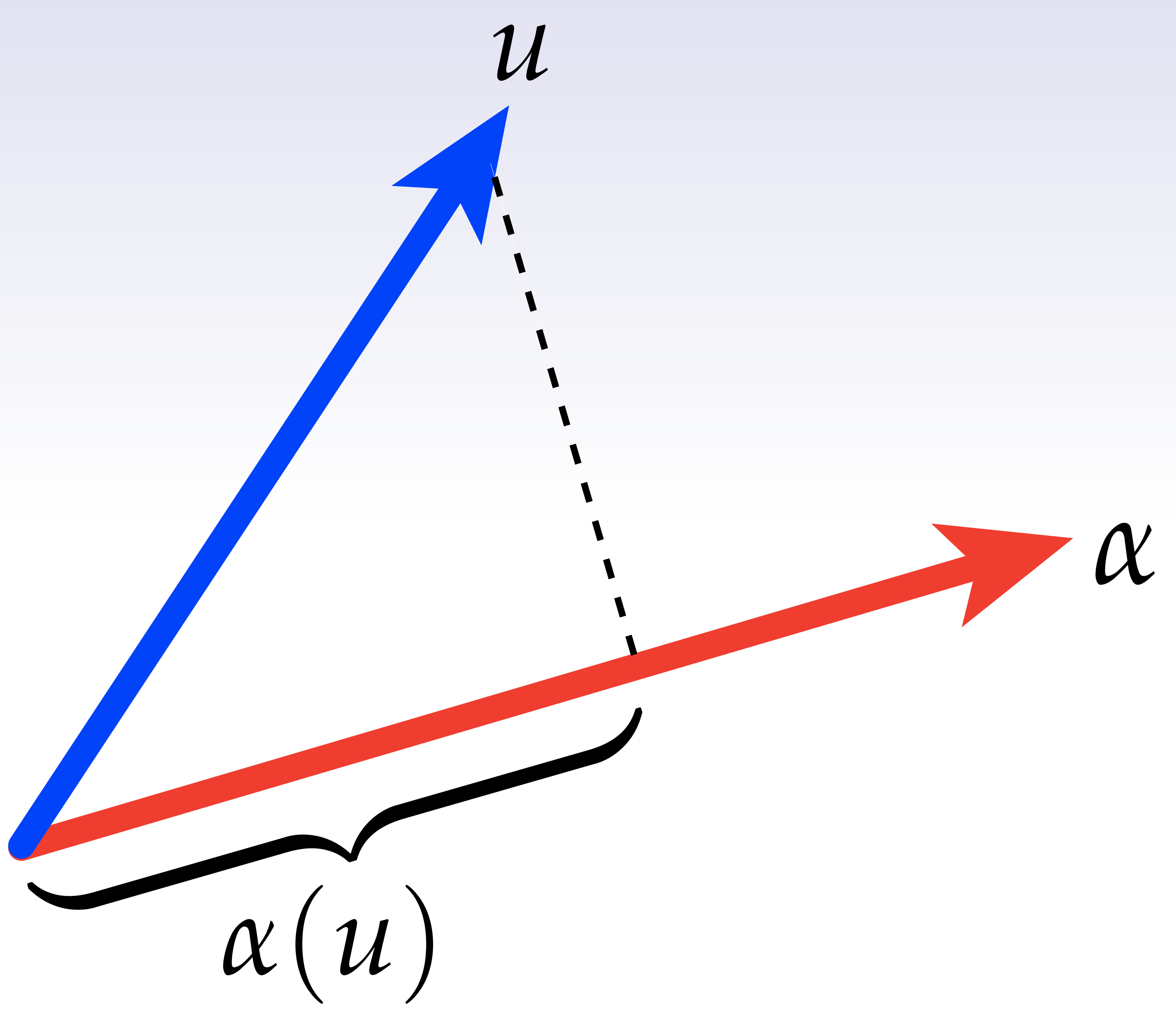
Differential Forms



Rows & Columns

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

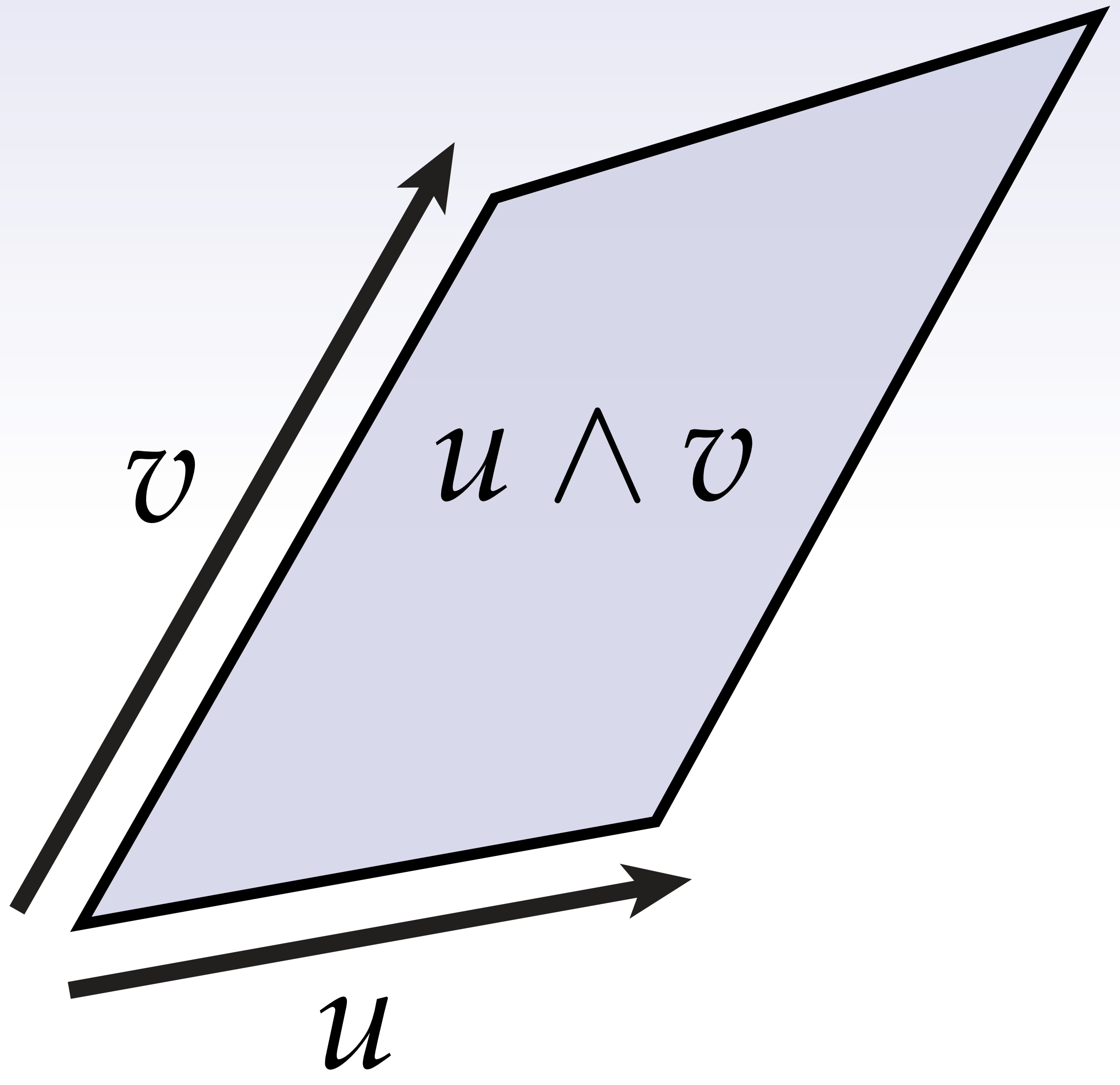
1-Vectors and 1-Forms



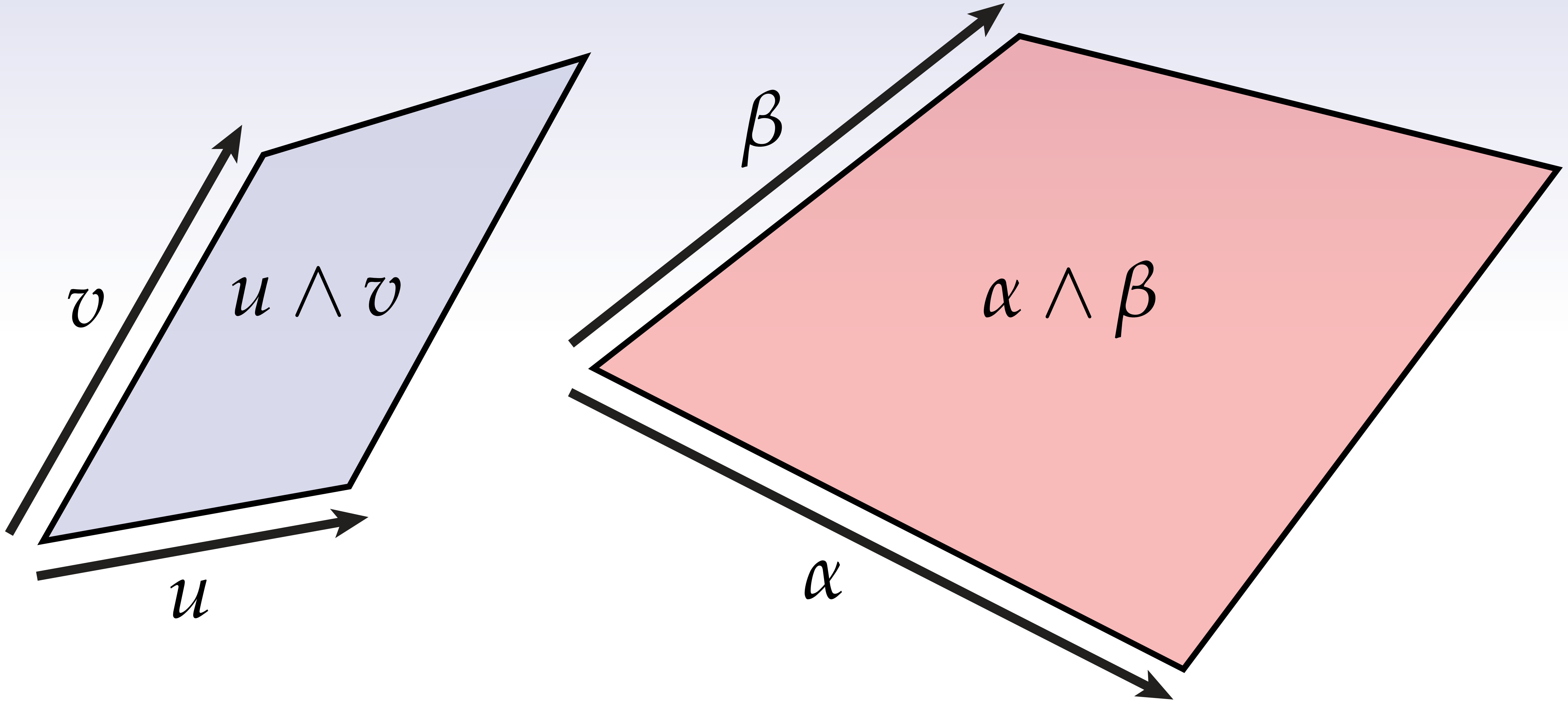
Analogy: *row vs. column vector*

2-Vectors and 2-Forms

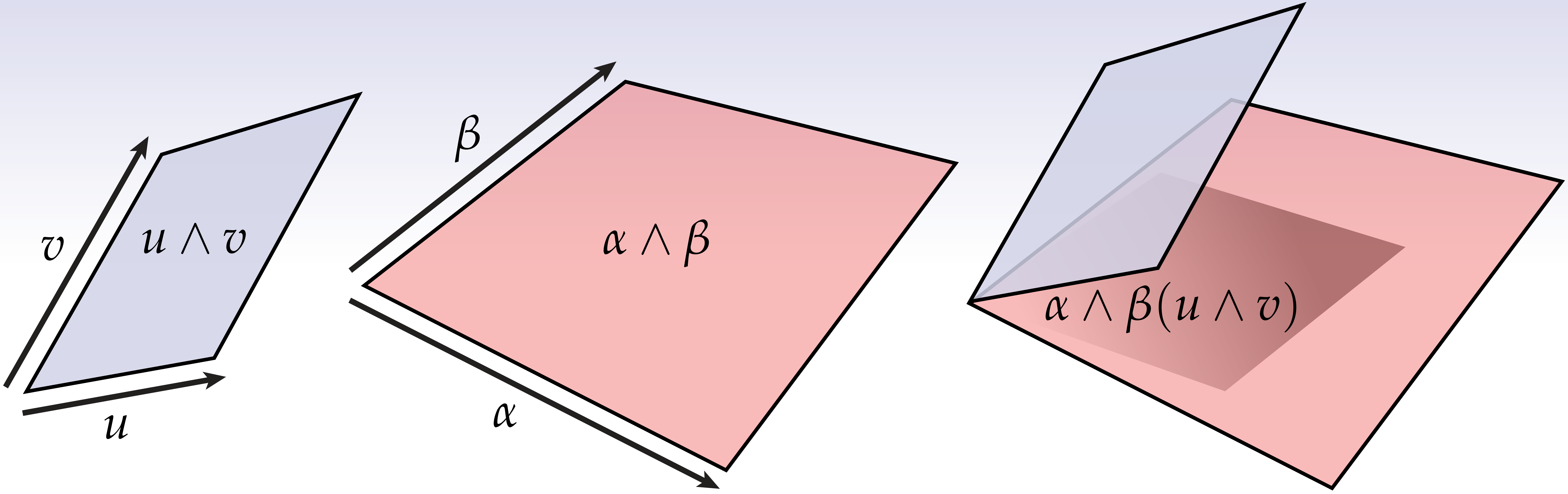
2-Vectors and 2-Forms



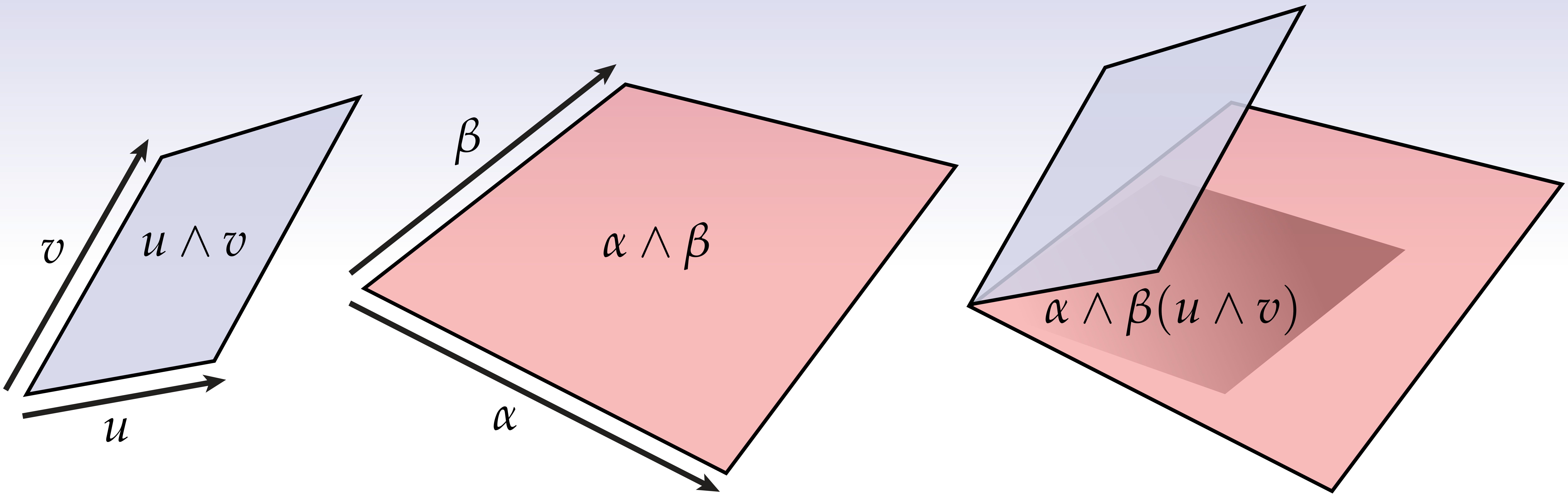
2-Vectors and 2-Forms



2-Vectors and 2-Forms

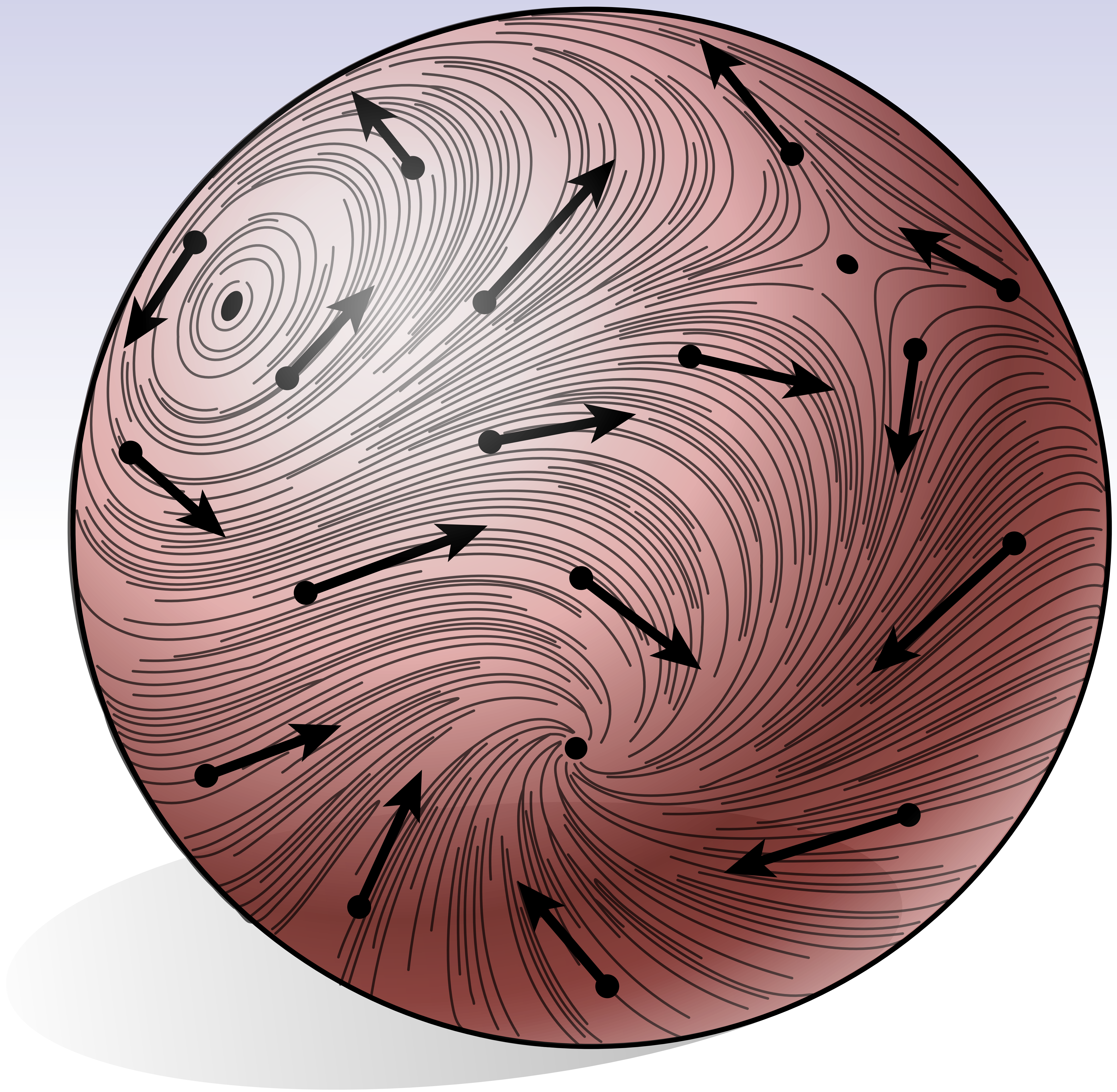


2-Vectors and 2-Forms

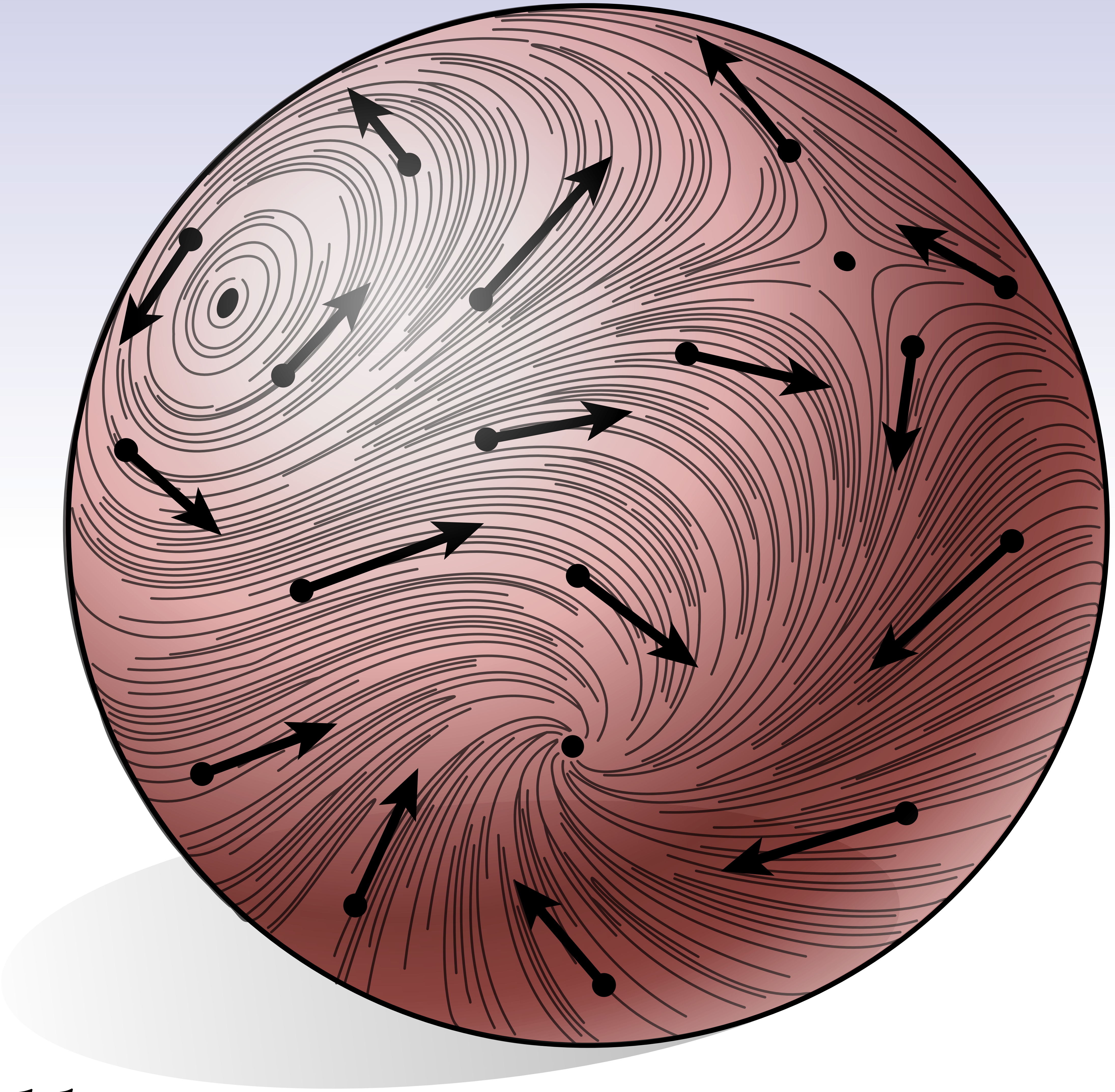


$$\alpha \wedge \beta(u \wedge v) = \alpha(u)\beta(v) - \alpha(v)\beta(u)$$

Differential 1-Form



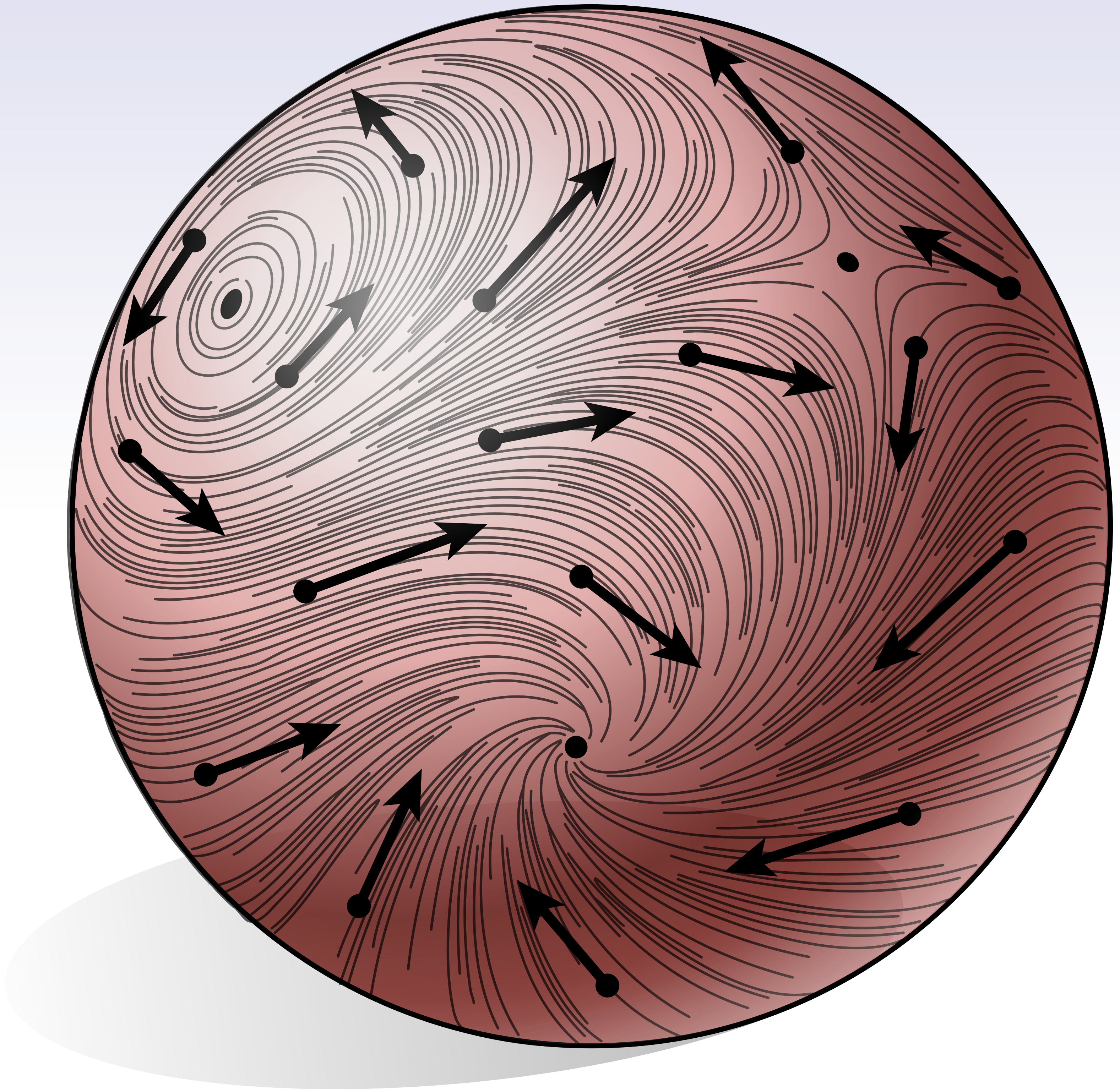
Differential 1-Form



Analogy: *vector field*

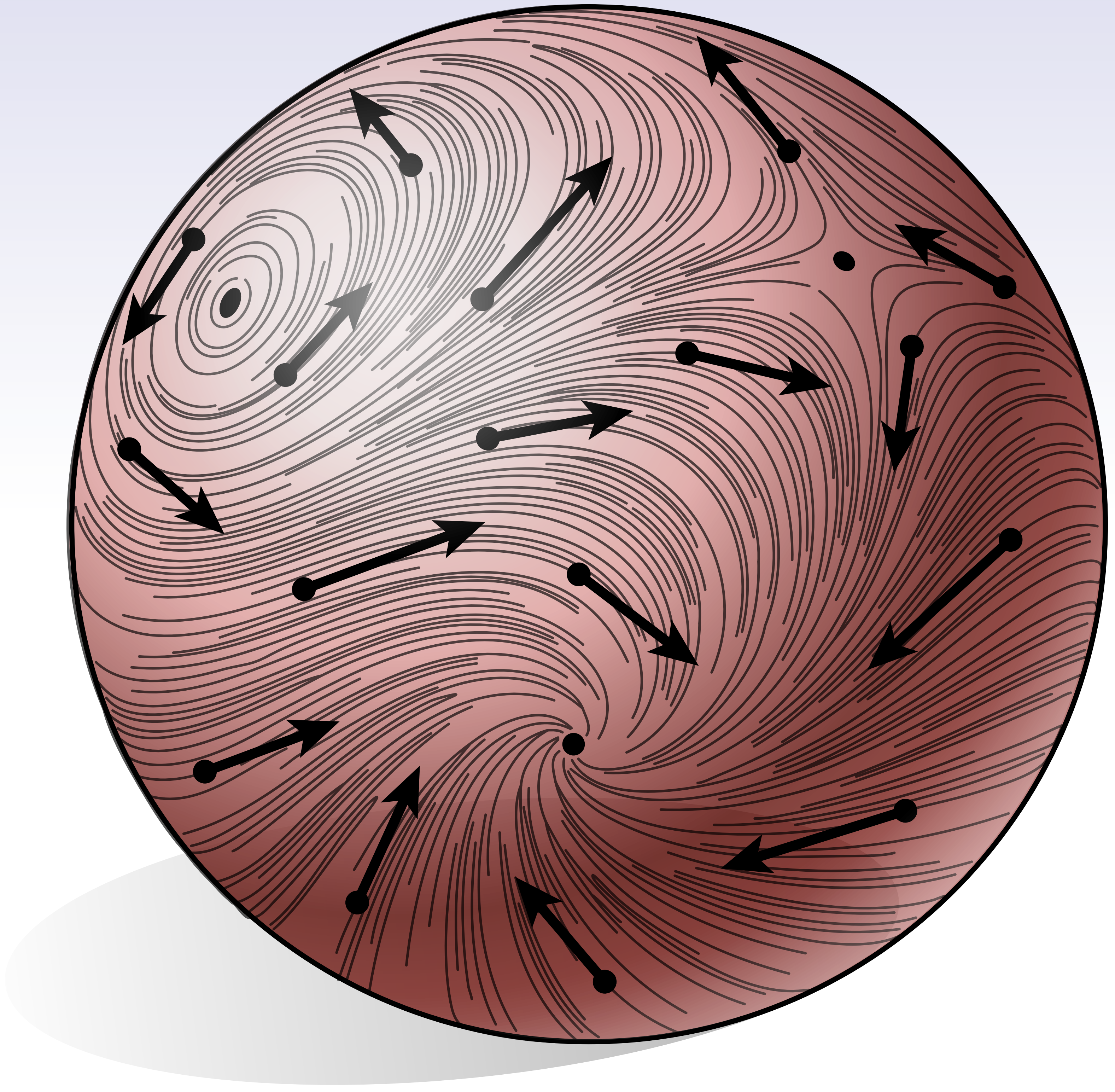
Differential 1-Forms & Vector Fields

Differential 1-Forms & Vector Fields

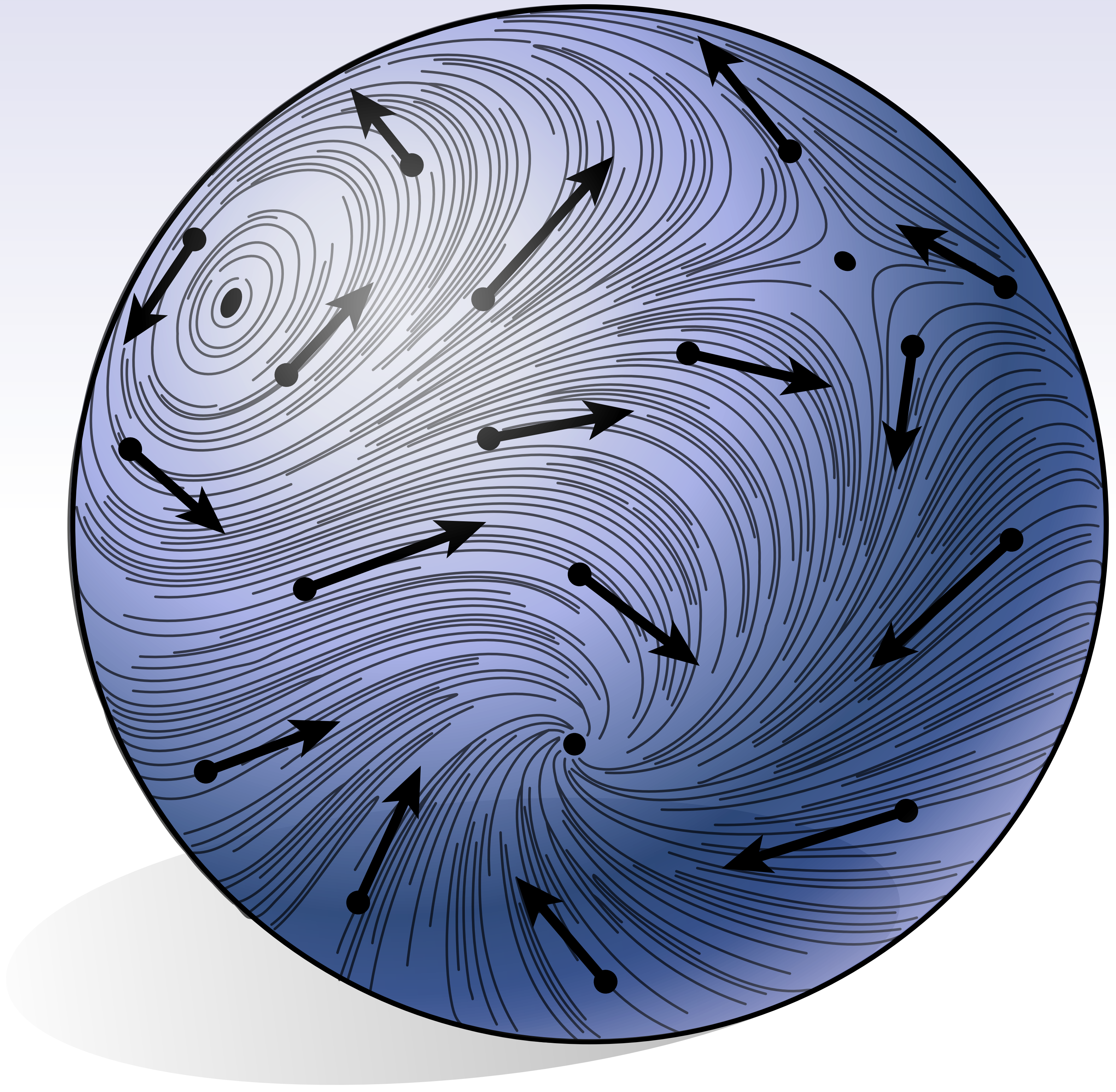


1-form

Differential 1-Forms & Vector Fields

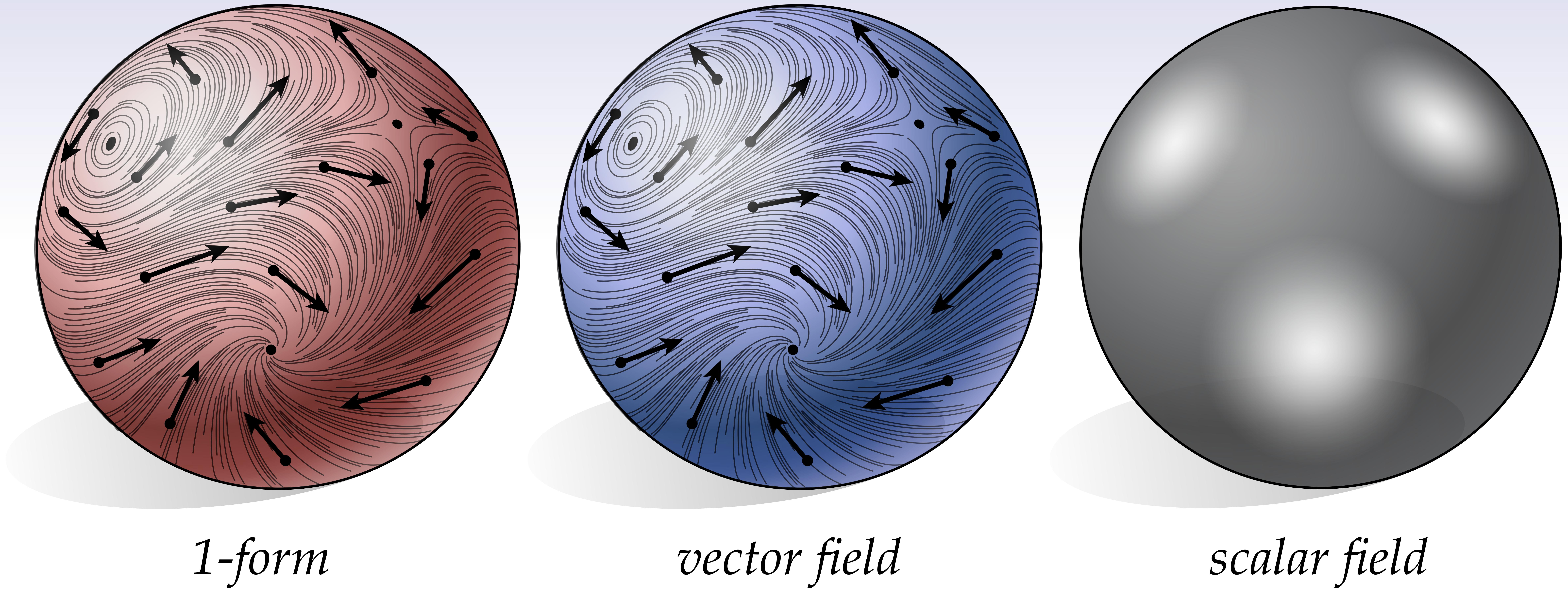


1-form

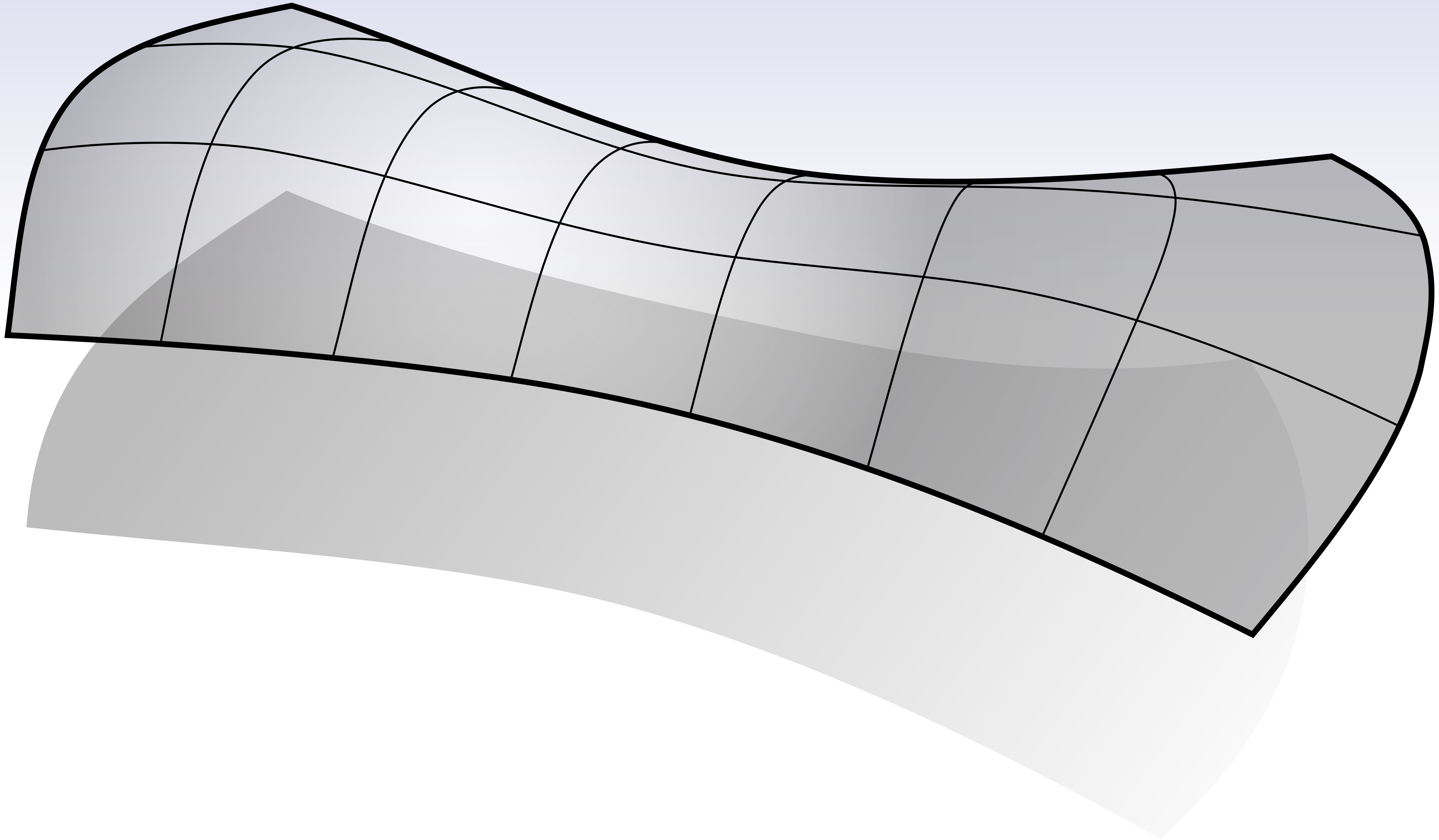


vector field

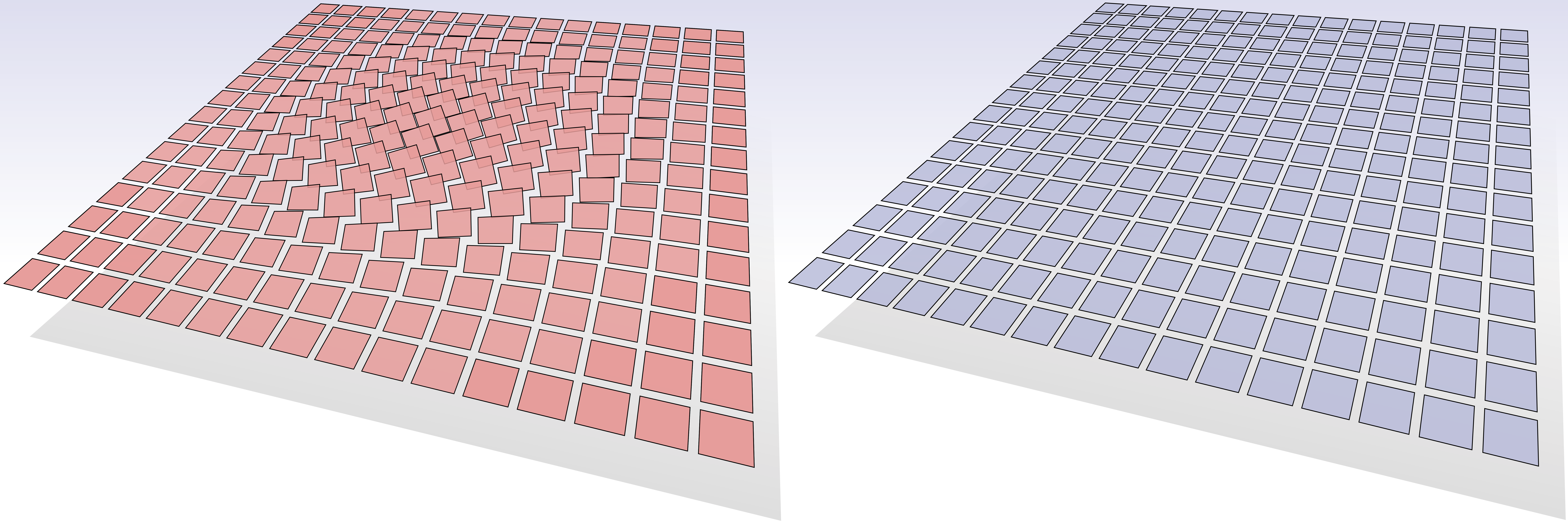
Differential 1-Forms & Vector Fields



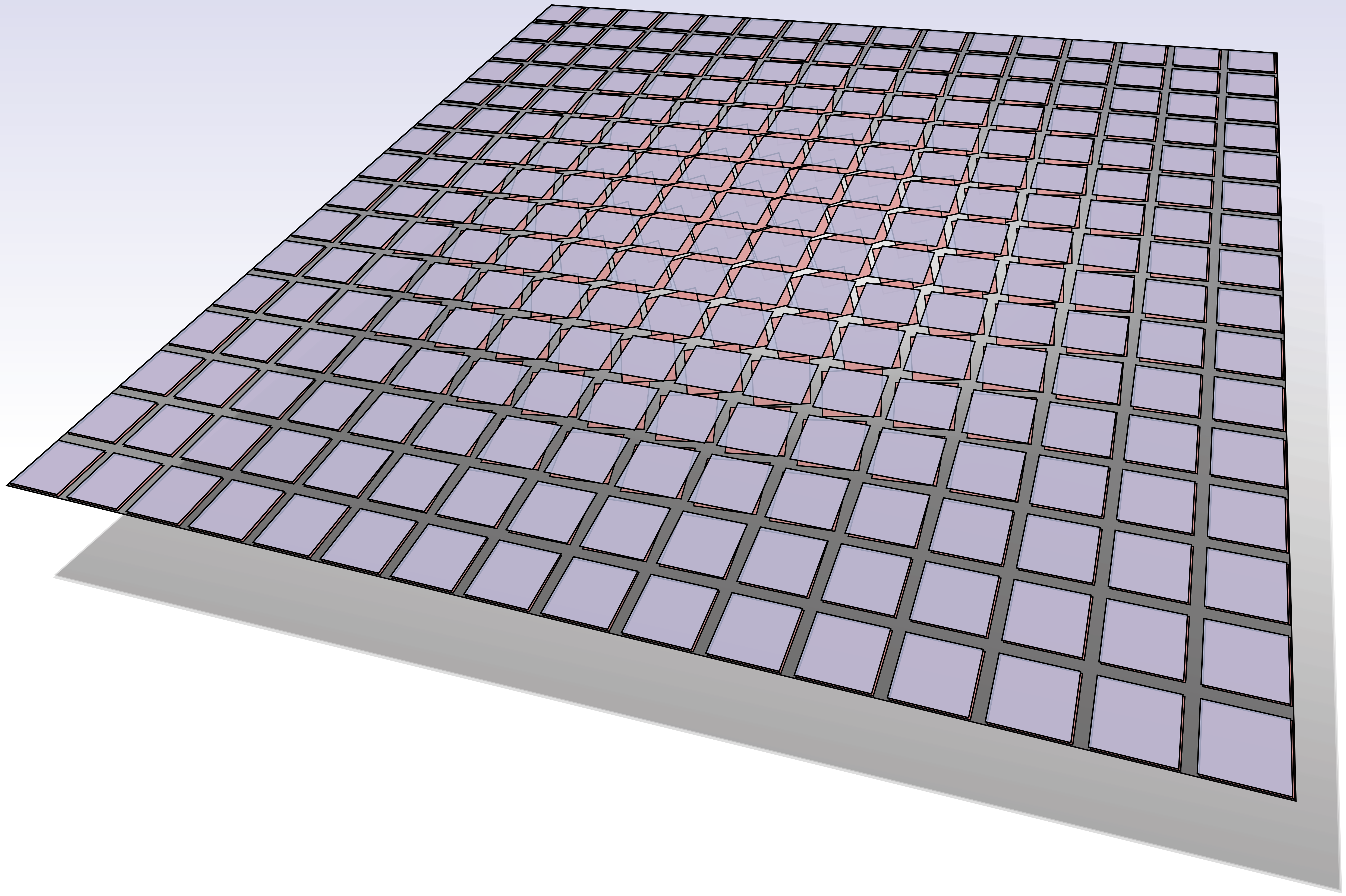
Scalar Fields & Differential 0-Forms



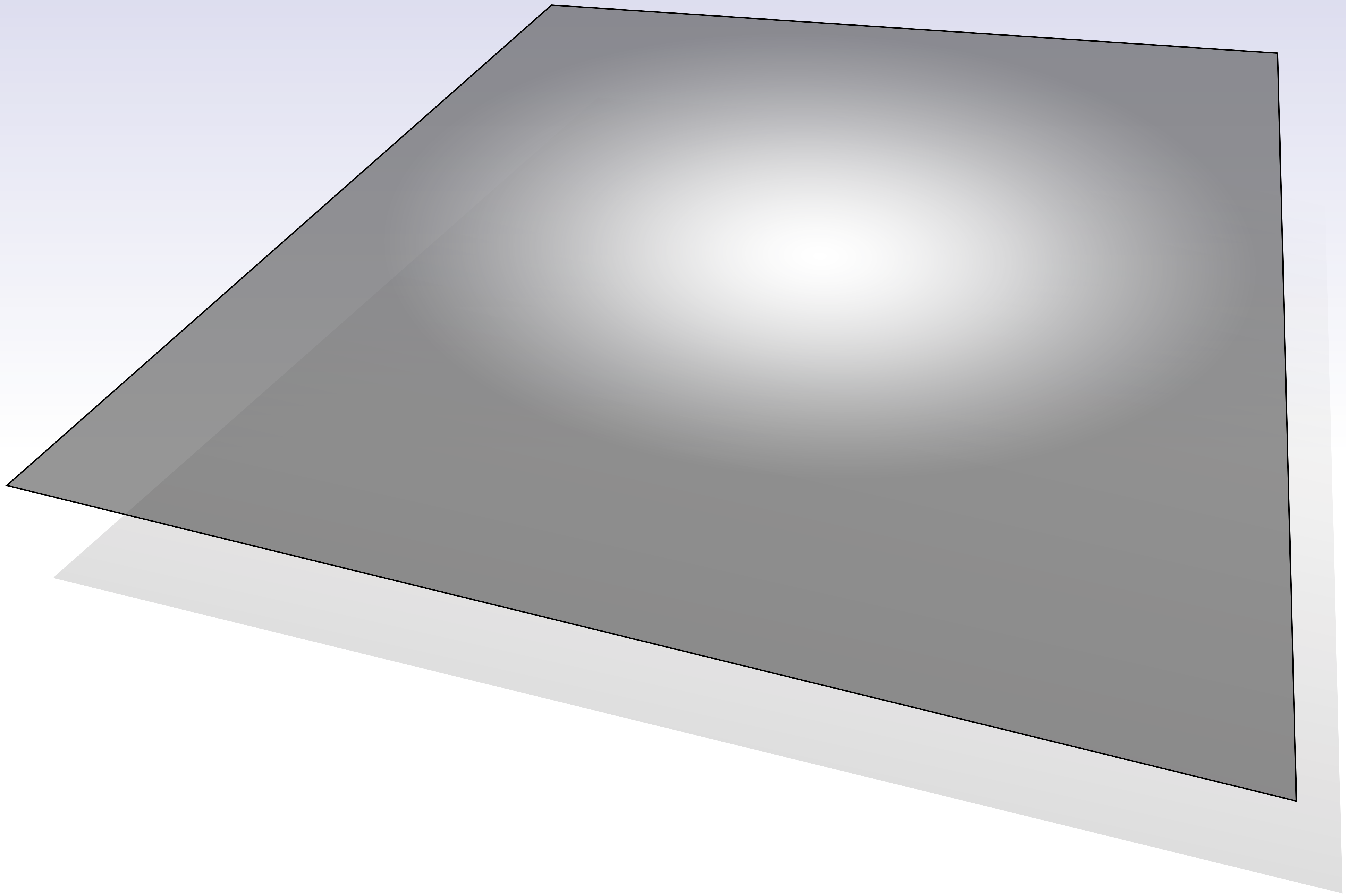
Differential 2-Forms



Differential 2-Forms



Differential 2-Forms



Sharp and Flat

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\quad\mathrm{T}\quad}$$

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

u, v

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u, v \xrightarrow{\flat}$$

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u, v \xrightarrow{\flat} u^\flat(v)$$

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u, v \xrightarrow{\flat} u^\flat(v)$$

$$\alpha, \beta$$

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u, v \xrightarrow{\flat} u^\flat(v)$$

$$\alpha, \beta \xrightarrow{\sharp}$$

Analogy: *transpose*

Sharp and Flat

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \xrightarrow{\mathsf{T}} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u, v \xrightarrow{\flat} u^\flat(v)$$

$$\alpha, \beta \xrightarrow{\sharp} \alpha(\beta^\sharp)$$

Analogy: *transpose*

Sharp and Flat w/ Mass

$$\begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Analogy: *transpose, mass matrix*

Sharp and Flat w/ Mass

$$\begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u^\flat(v) = u^\mathsf{T} M v$$

Analogy: *transpose, mass matrix*

Sharp and Flat w/ Mass

$$\begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u^\flat(v) = u^\mathsf{T} M v$$

$$\alpha(\beta^\sharp) =$$

Analogy: *transpose, mass matrix*

Sharp and Flat w/ Mass

$$\begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u^\flat(v) = u^\mathsf{T} M v$$

$$\alpha(\beta^\sharp) = \alpha M^{-1} \beta^\mathsf{T}$$

Analogy: *transpose, mass matrix*

Sharp and Flat w/ Mass

$$\begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u^\flat(v) = u^\mathsf{T} M v \qquad \Longleftrightarrow \qquad u^\flat(\cdot) = \langle u, \cdot \rangle$$

$$\alpha(\beta^\sharp) = \alpha M^{-1} \beta^\mathsf{T}$$

Analogy: *transpose, mass matrix*

Sharp and Flat w/ Mass

$$\begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$u^\flat(v) = u^\mathsf{T} M v \qquad \Longleftrightarrow \qquad u^\flat(\cdot) = \langle u, \cdot \rangle$$

$$\alpha(\beta^\sharp) = \alpha M^{-1} \beta^\mathsf{T} \qquad \Longleftrightarrow \qquad \langle \alpha^\sharp, \cdot \rangle = \alpha(\cdot)$$

Analogy: *transpose, mass matrix*

Differential Forms - Summary

Differential Forms - Summary

- *primal* and *dual* exterior algebra

Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured

Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured
  - forms: take measurement

Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured
  - forms: take measurement
  - result is a scalar

Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured
  - forms: take measurement
  - result is a scalar
- define over surface

Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured
  - forms: take measurement
  - result is a scalar
- define over surface
  - gets measured: *k-vector field*

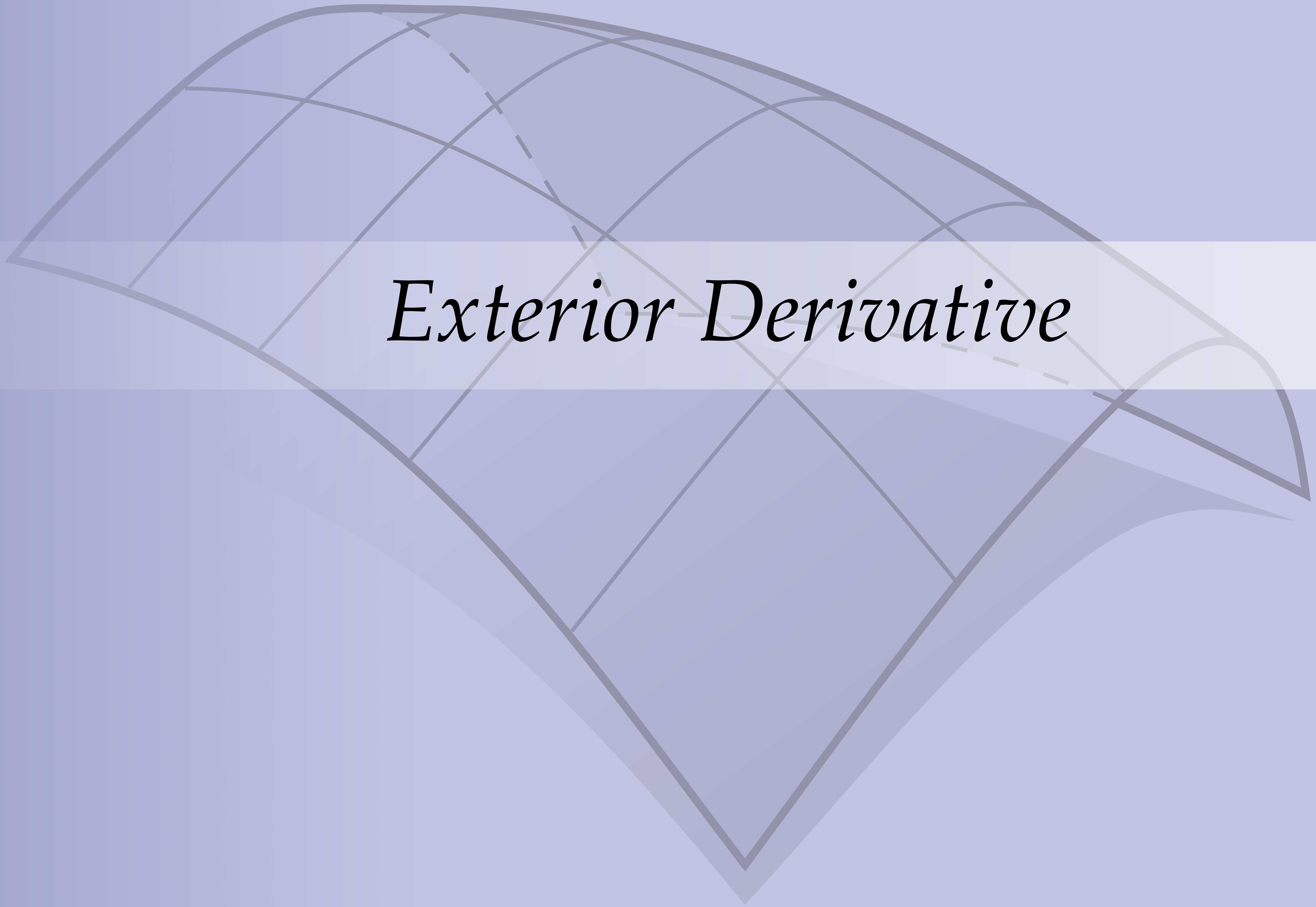
Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured
  - forms: take measurement
  - result is a scalar
- define over surface
  - gets measured: *k-vector field*
  - takes measurement: *differential k-form*

Differential Forms - Summary

- *primal* and *dual* exterior algebra
  - vectors: get measured
  - forms: take measurement
  - result is a scalar
- define over surface
  - gets measured: *k-vector field*
  - takes measurement: *differential k-form*
- convert between vectors and 1-forms w/ *sharp* & *flat*

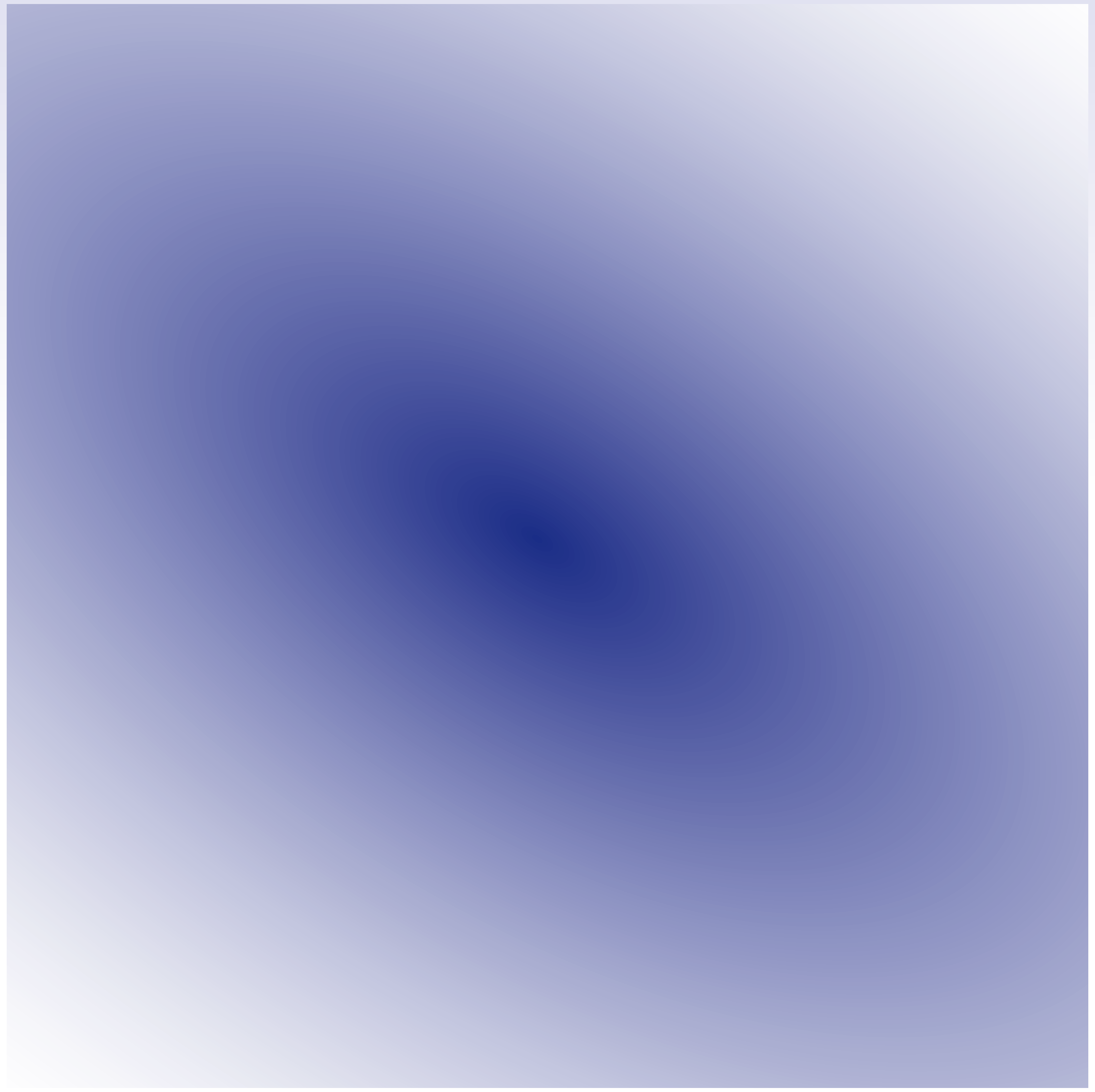
Exterior Derivative



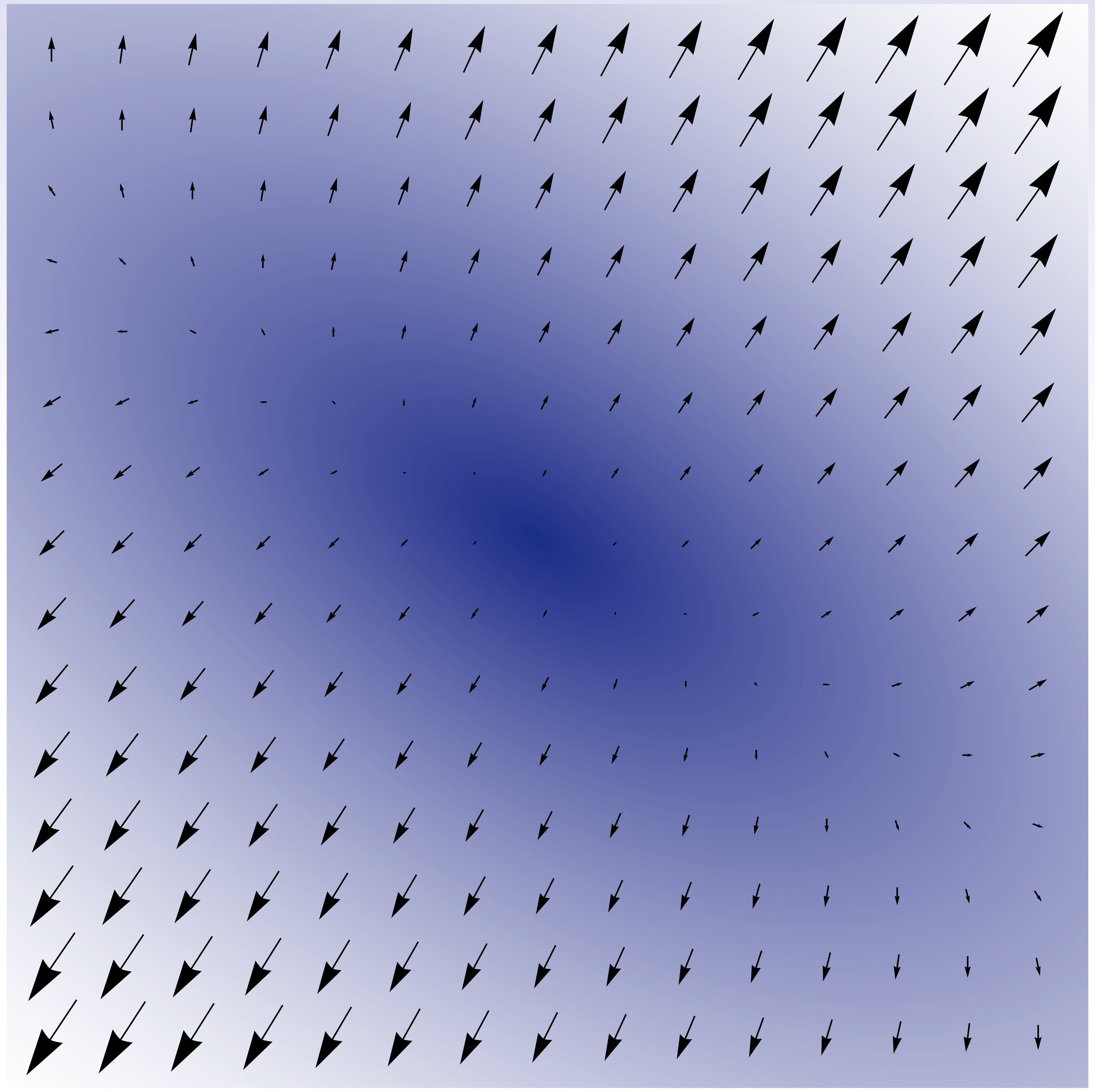
Review: Div, Grad, and Curl

Review: Div, Grad, and Curl

ϕ

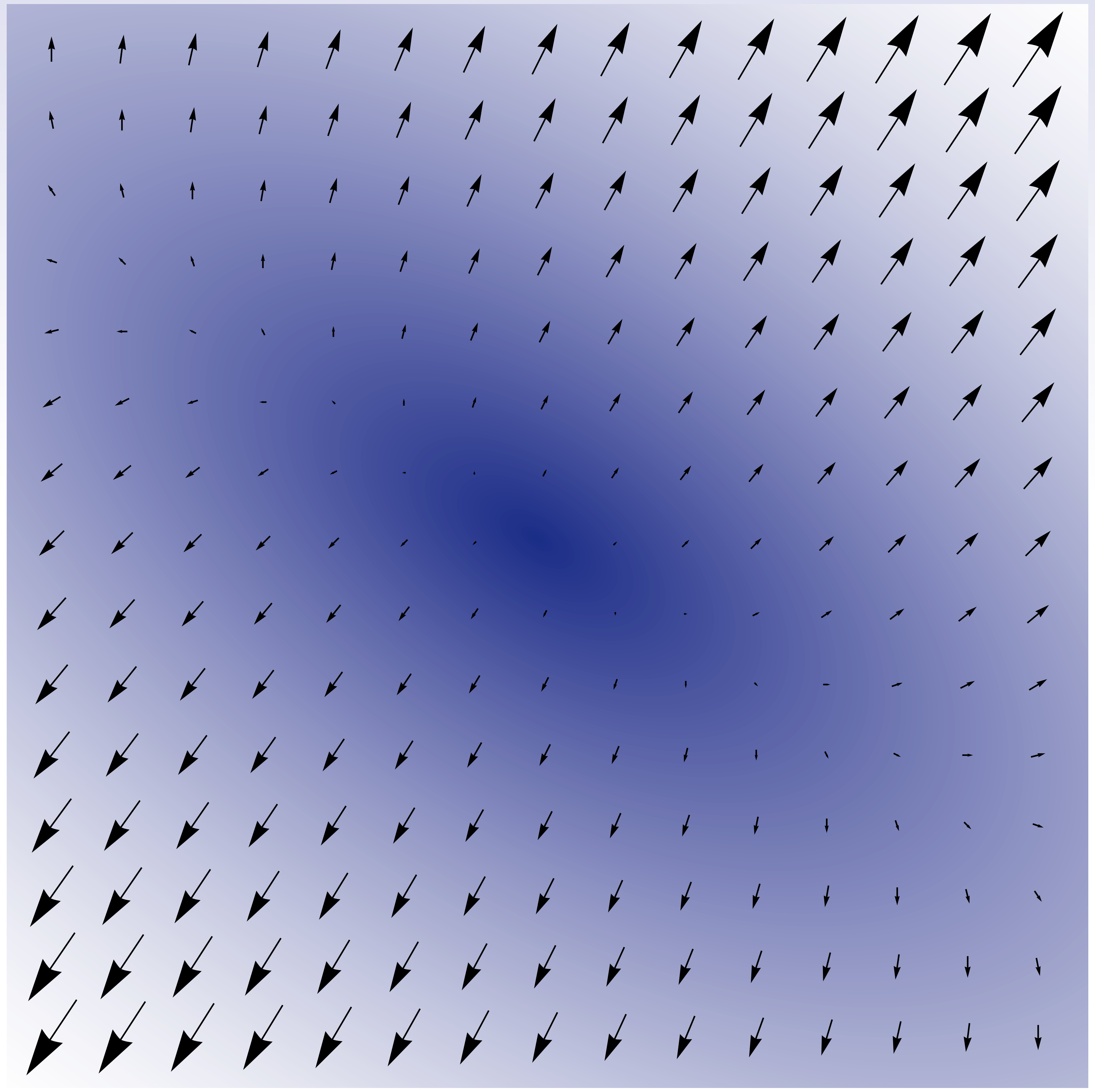


Review: Div, Grad, and Curl



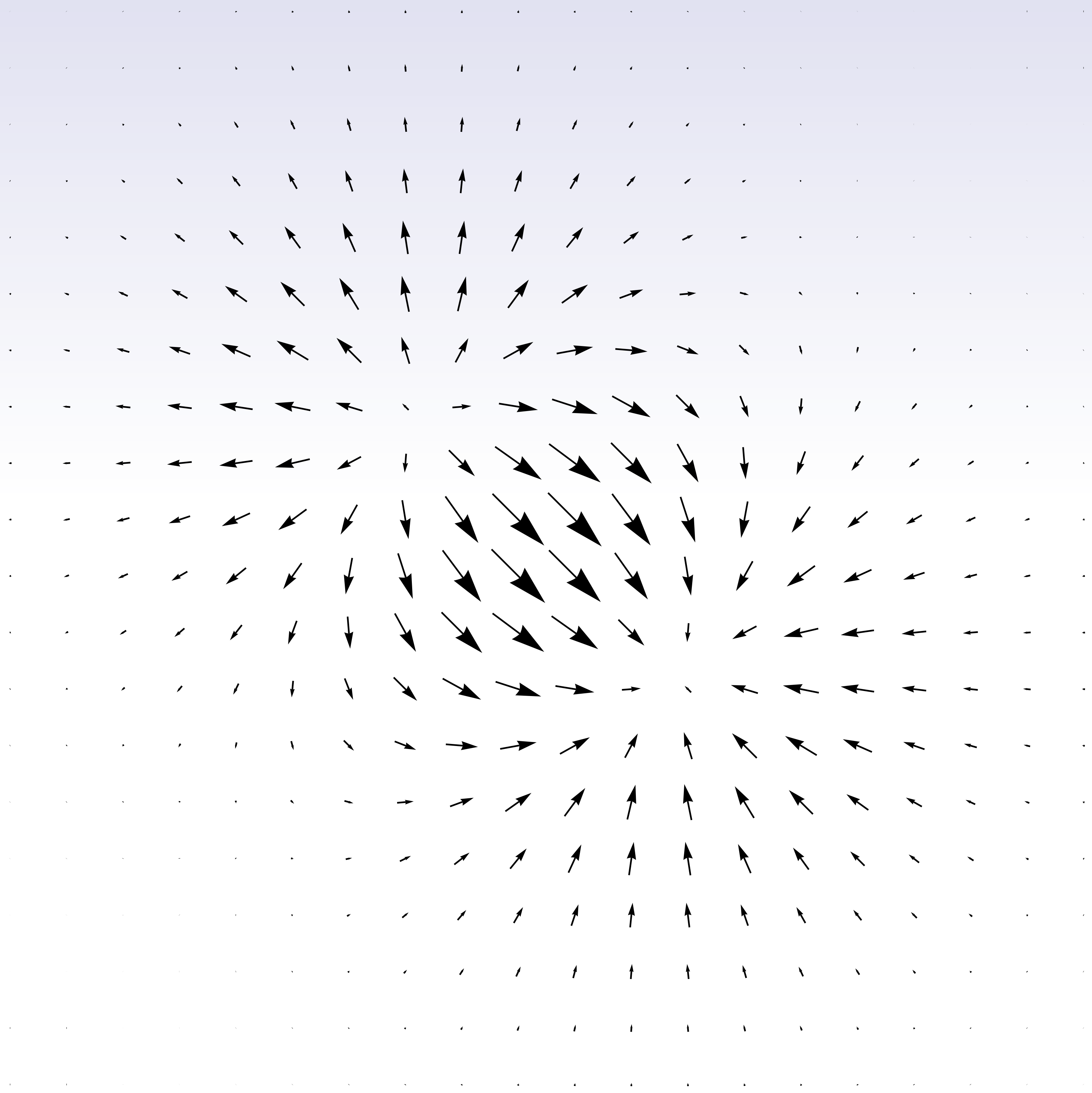
grad ϕ

Review: Div, Grad, and Curl

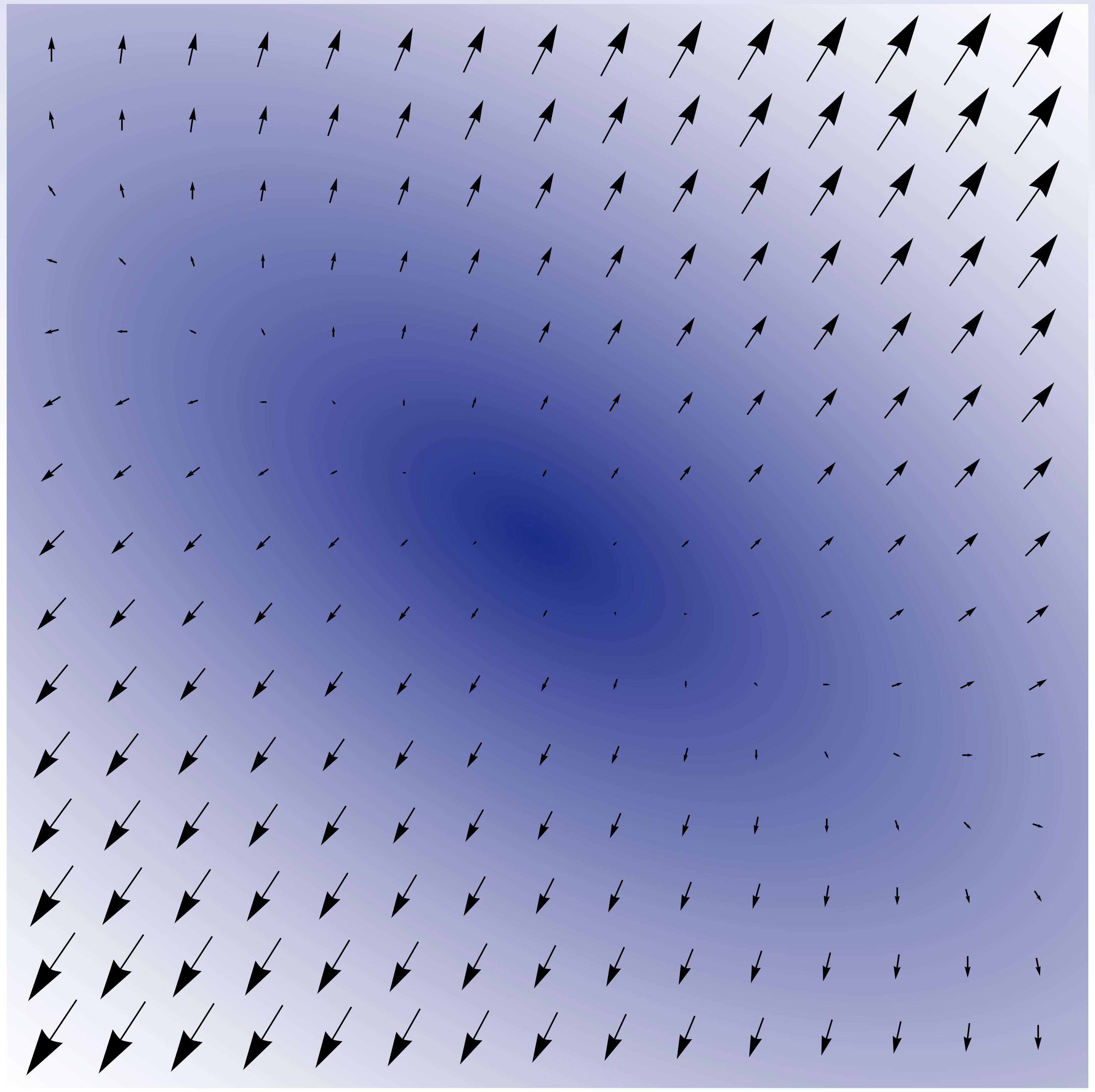


grad ϕ

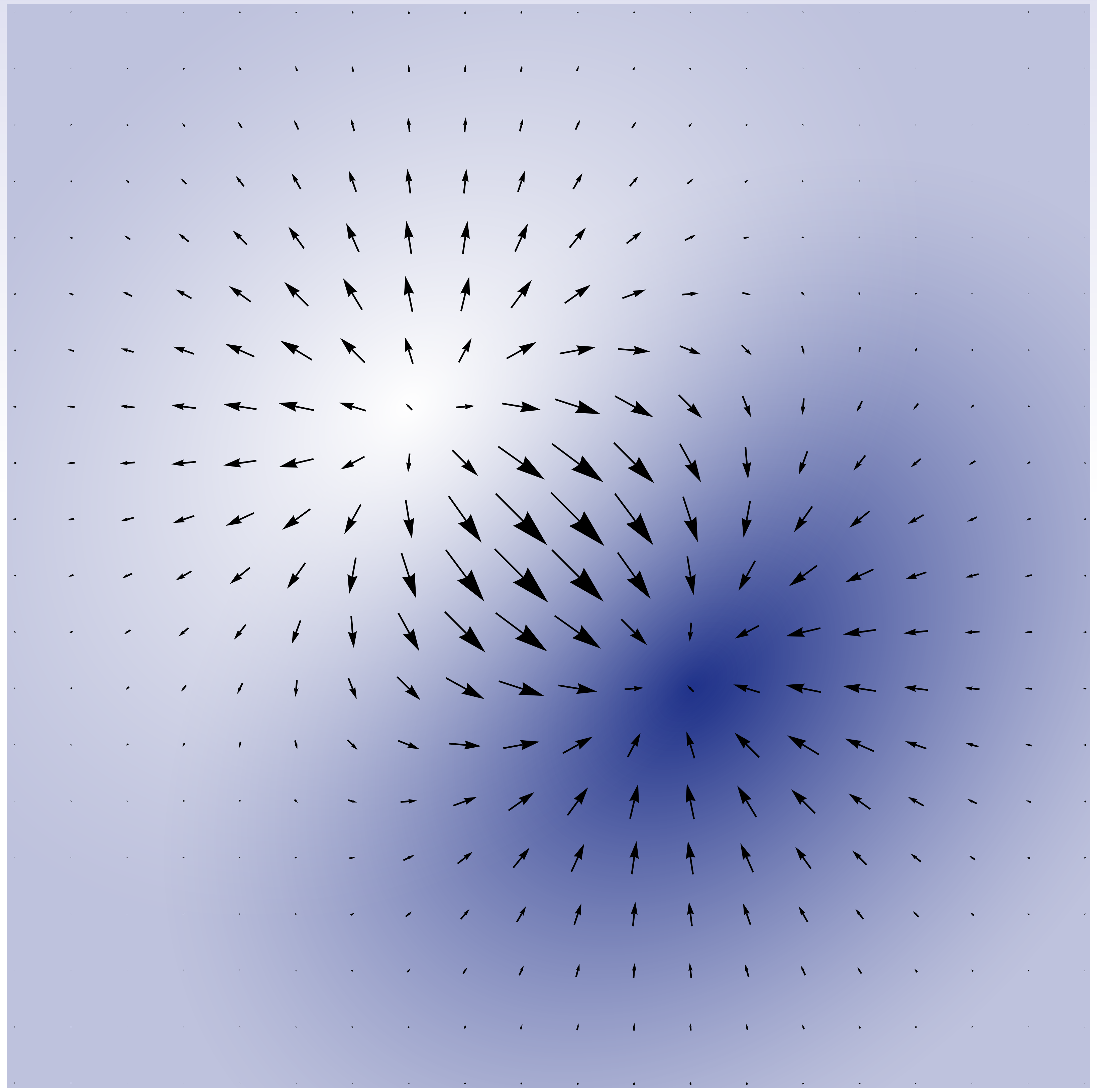
X



Review: Div, Grad, and Curl

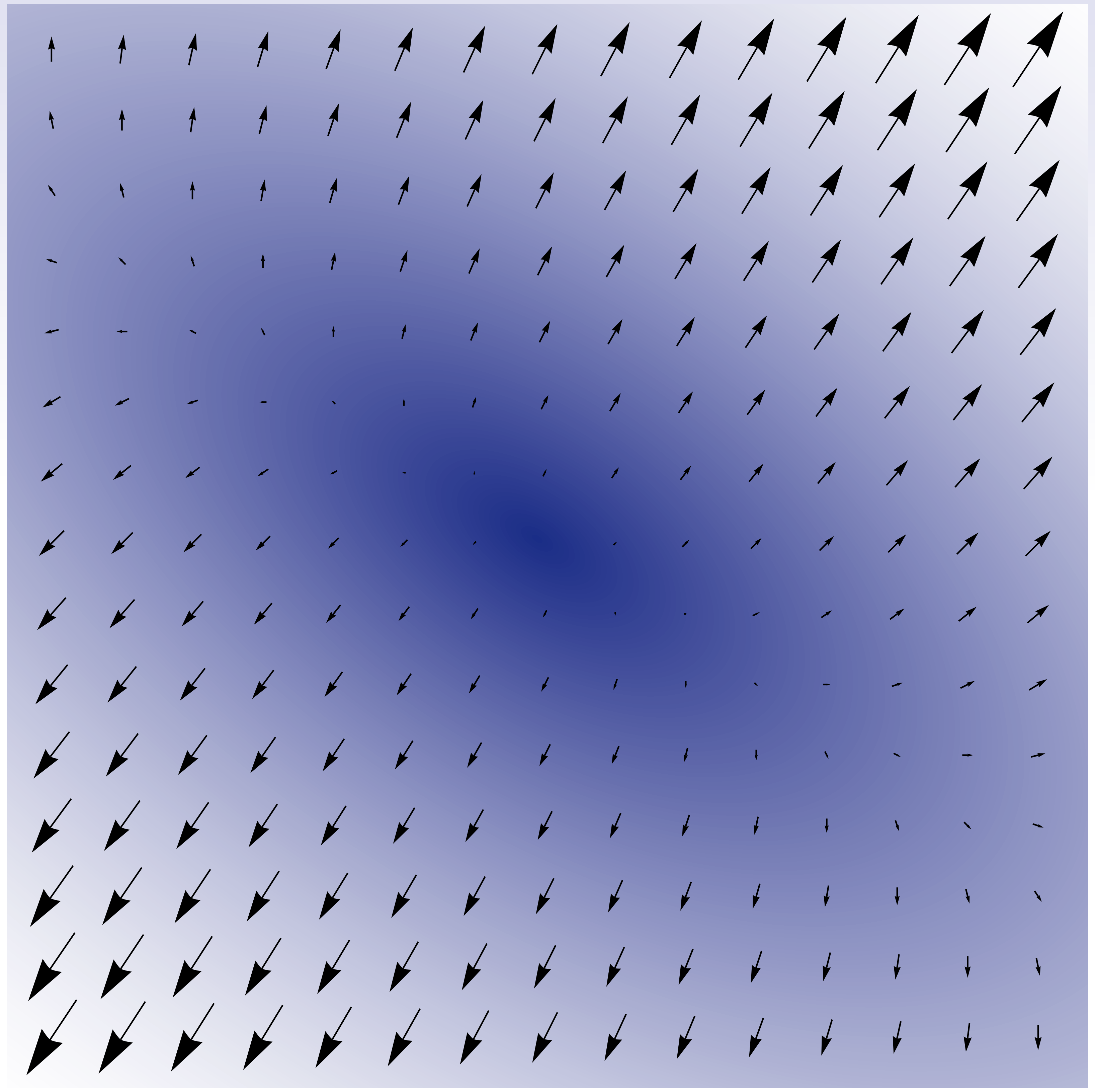


grad ϕ

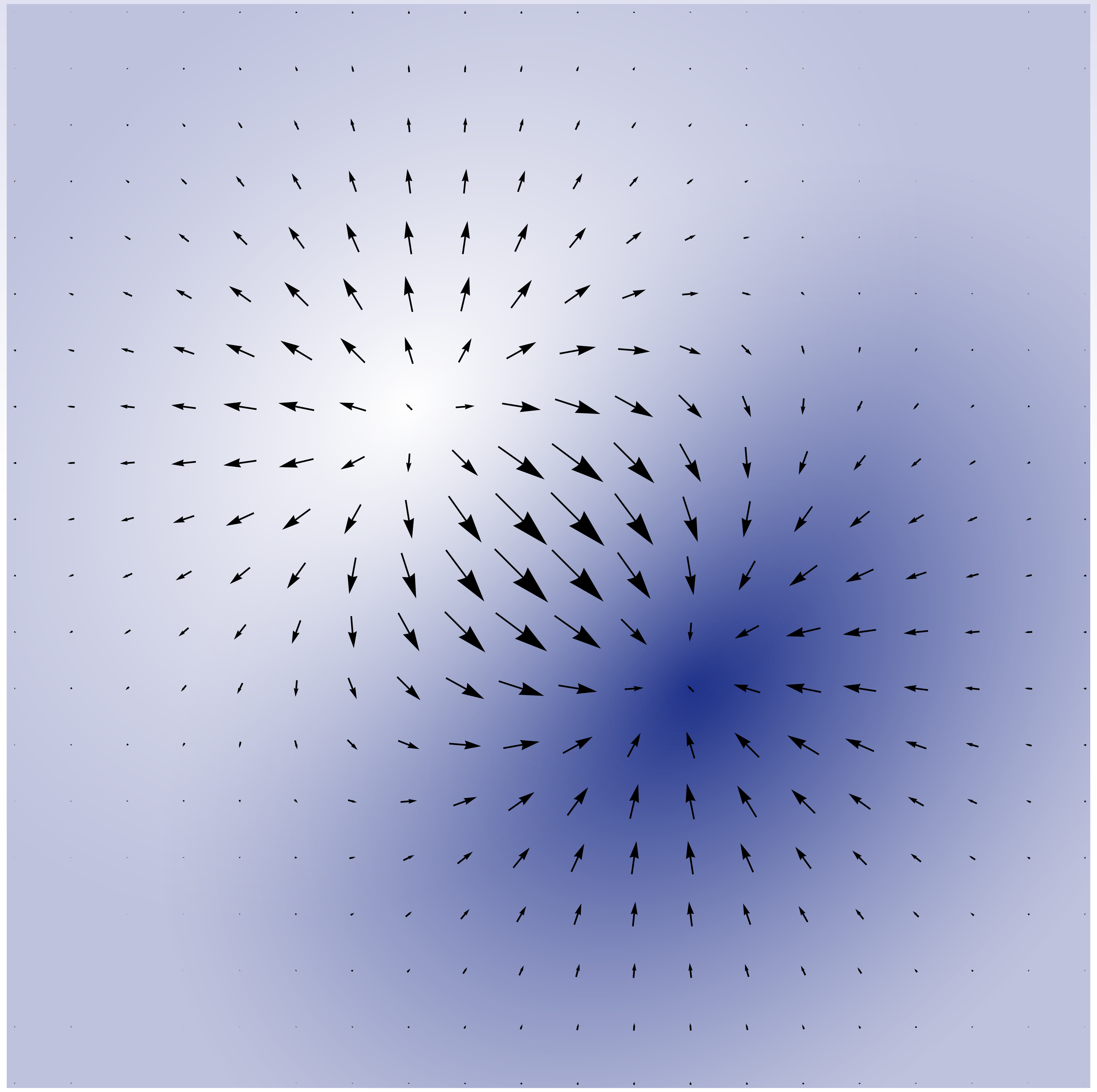


div X

Review: Div, Grad, and Curl

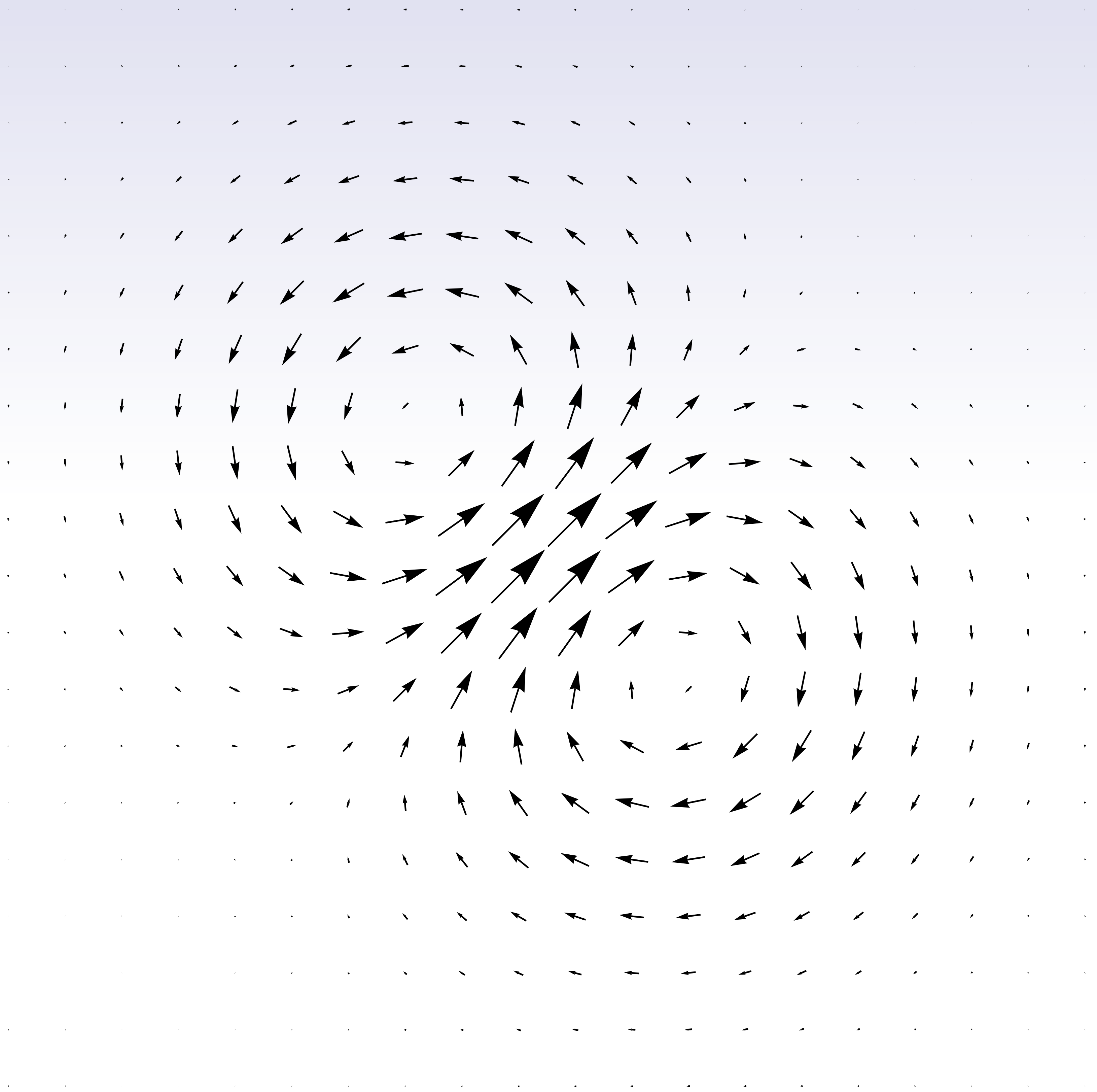


grad ϕ

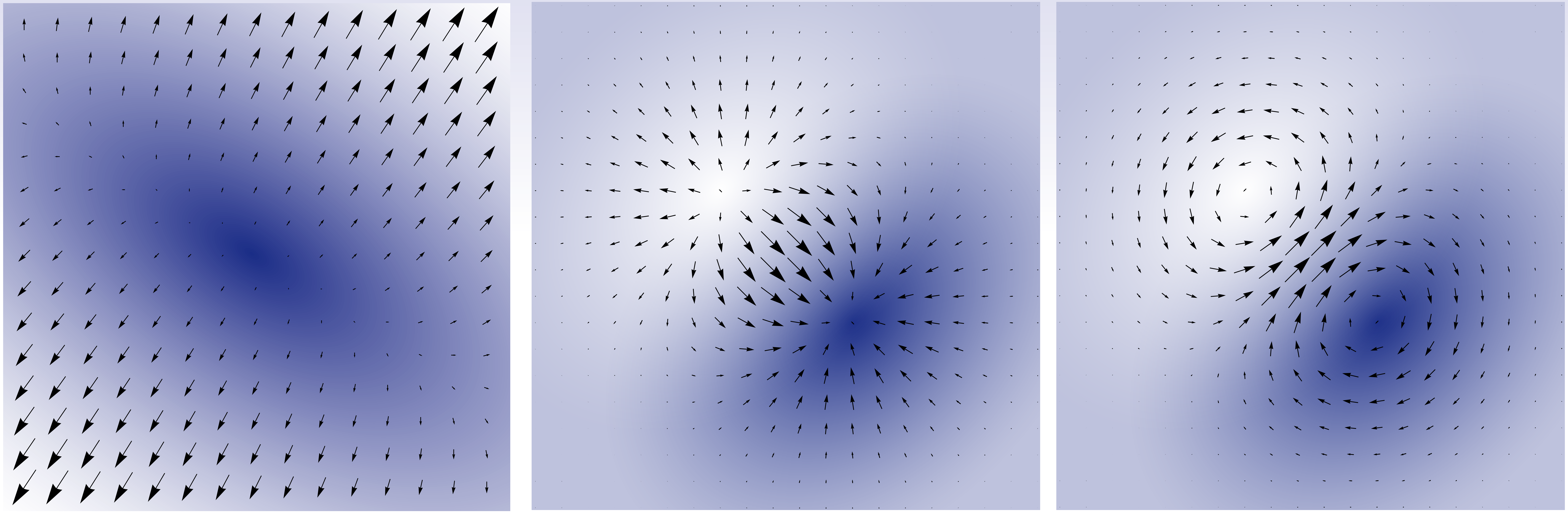


div X

Y

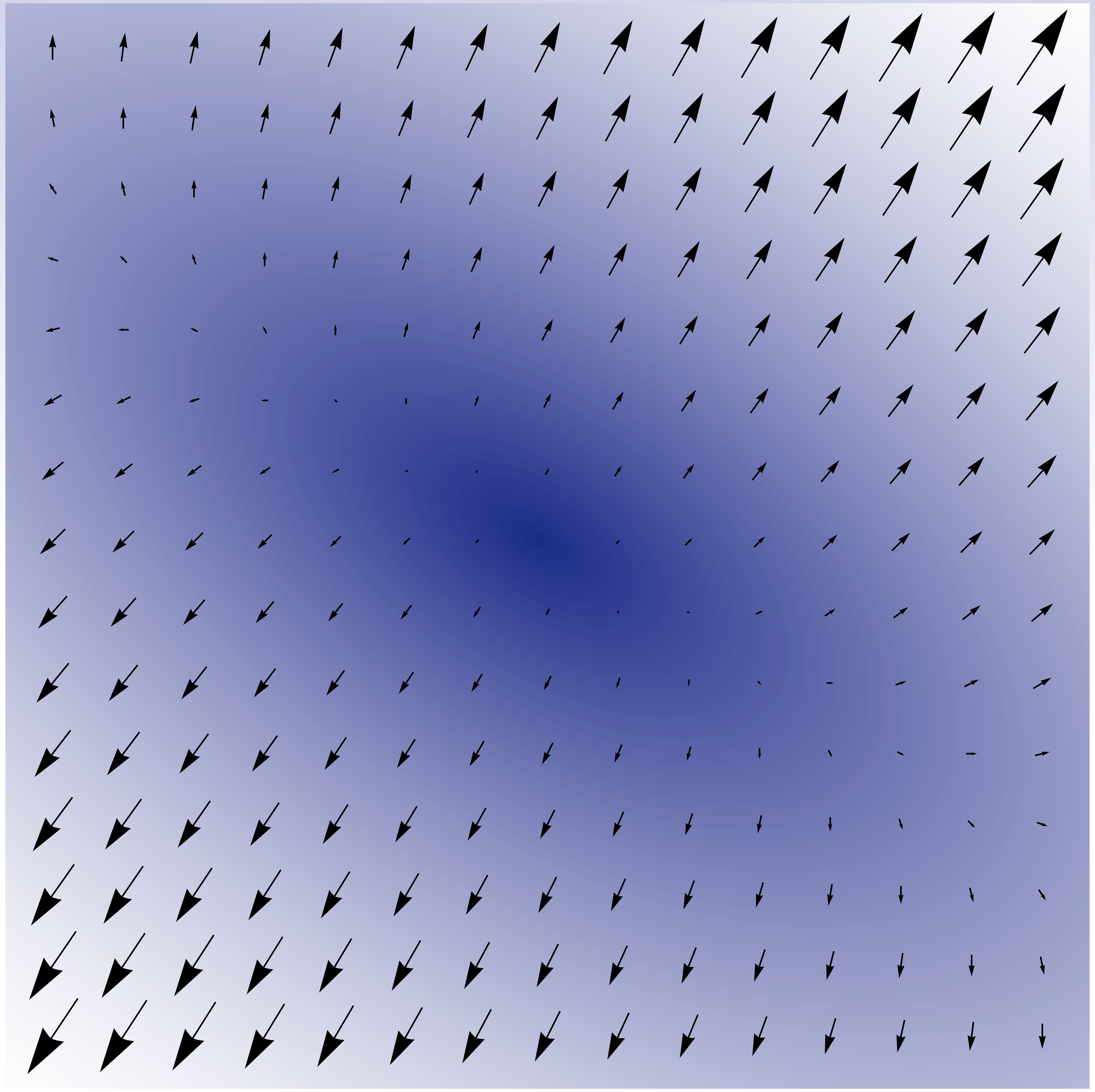


Review: Div, Grad, and Curl



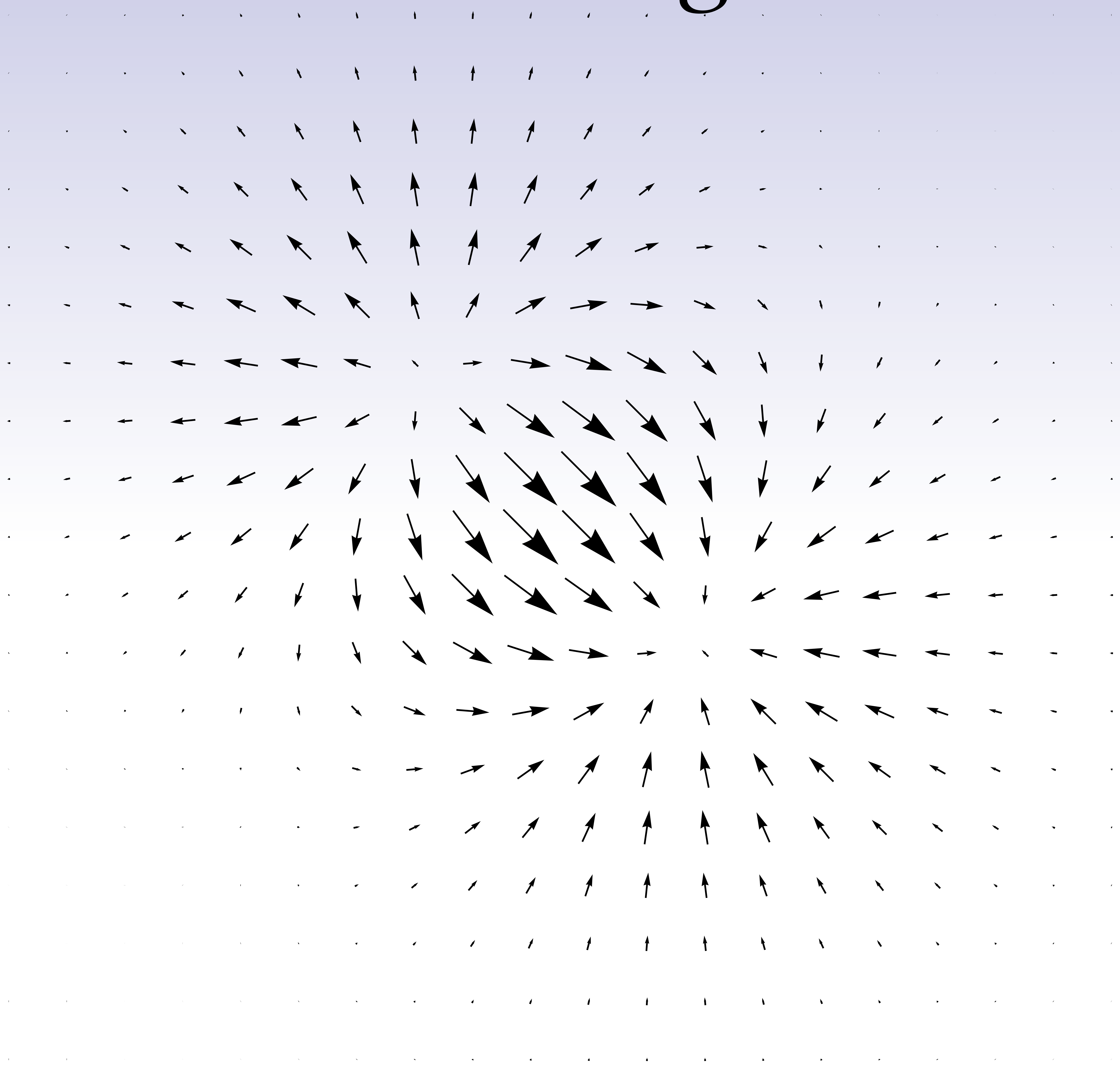
grad ϕ div X curl Y

Exterior Derivative - Gradient



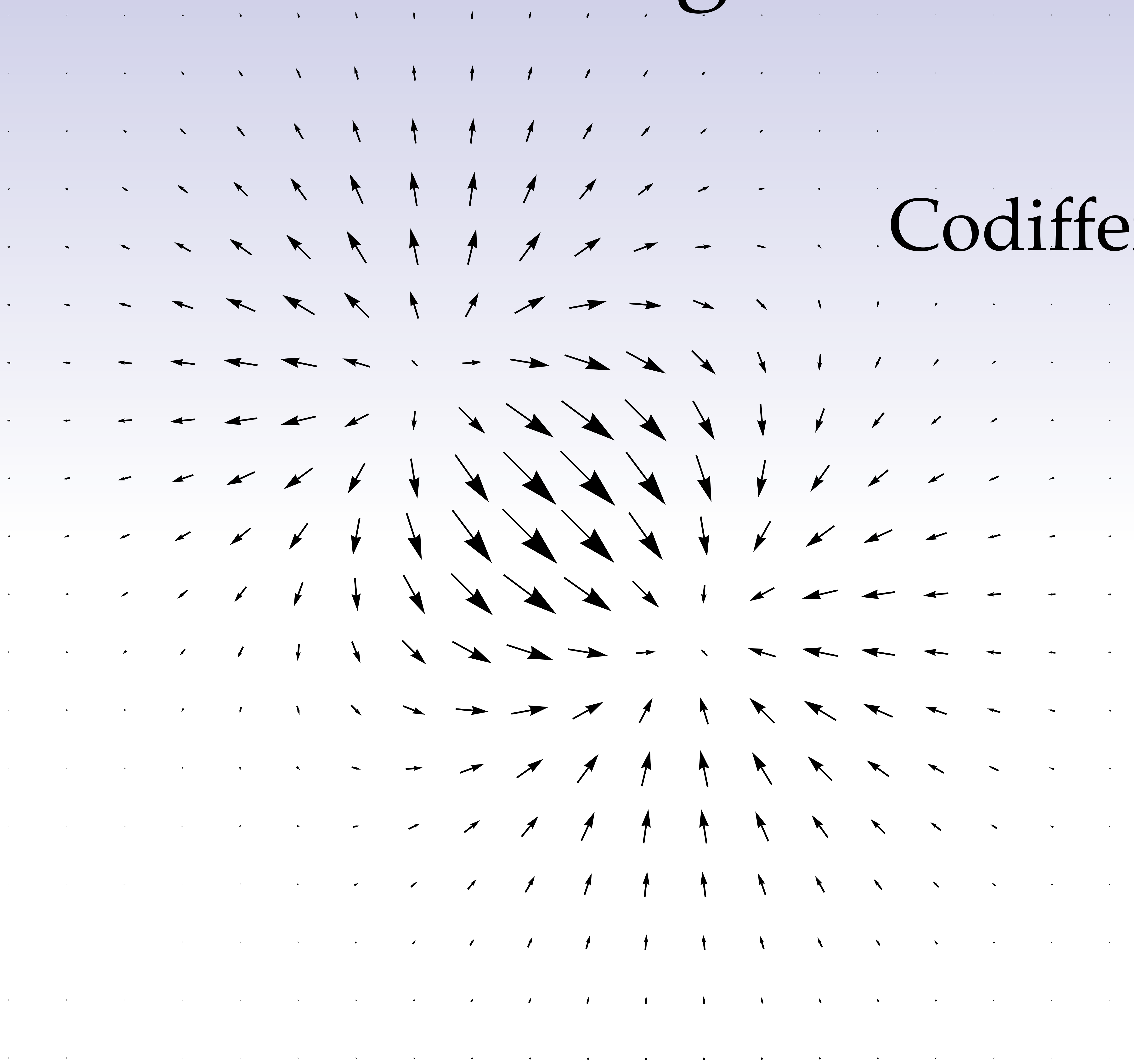
$$\nabla \phi = (d\phi)^\sharp$$

Exterior Derivative - Divergence



$$\nabla \cdot X = \star d \star X^\flat$$

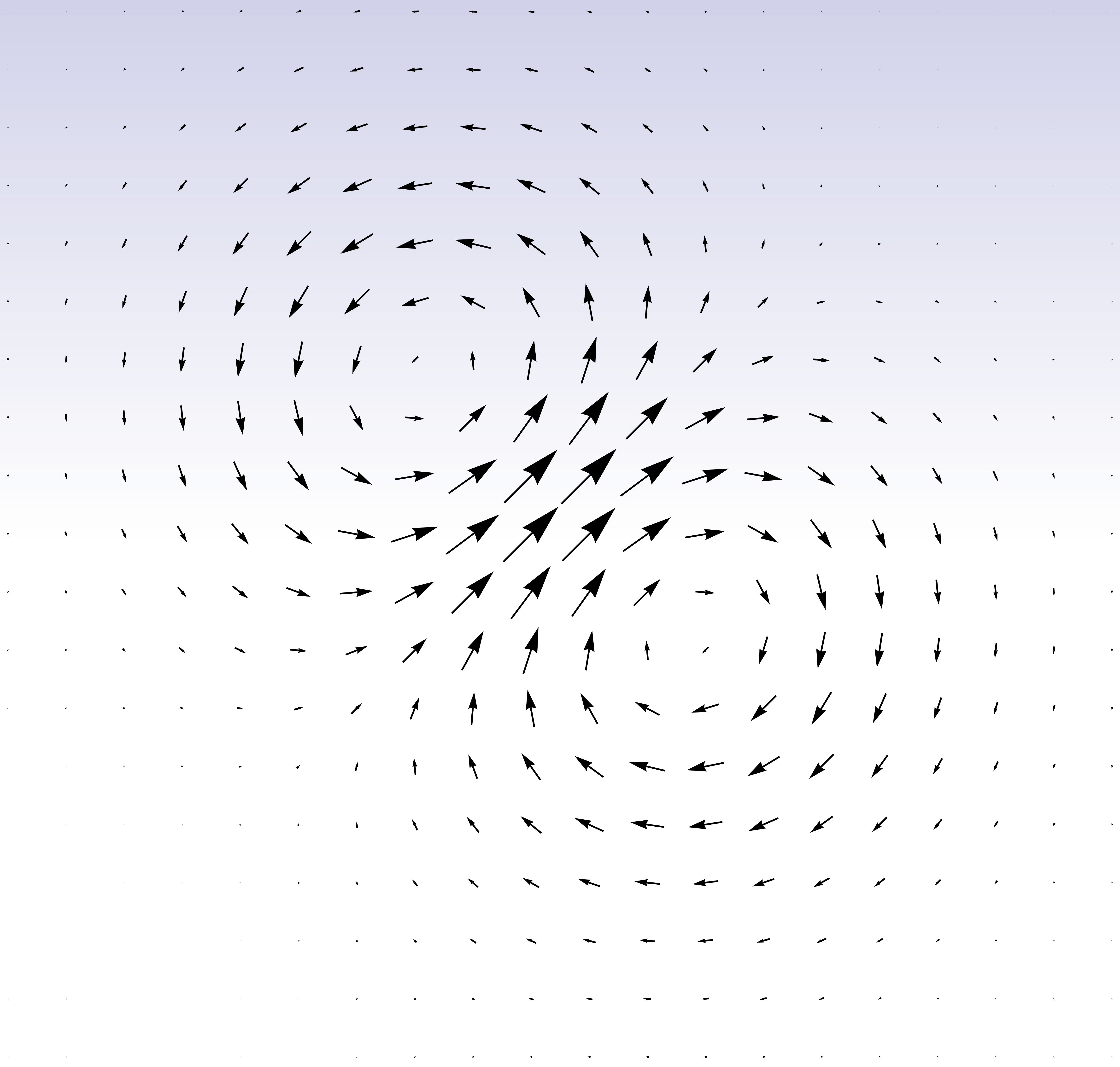
Exterior Derivative - Divergence



Codifferential: $\delta = \star d \star$

$$\nabla \cdot X = \star d \star X^\flat$$

Exterior Derivative - Curl



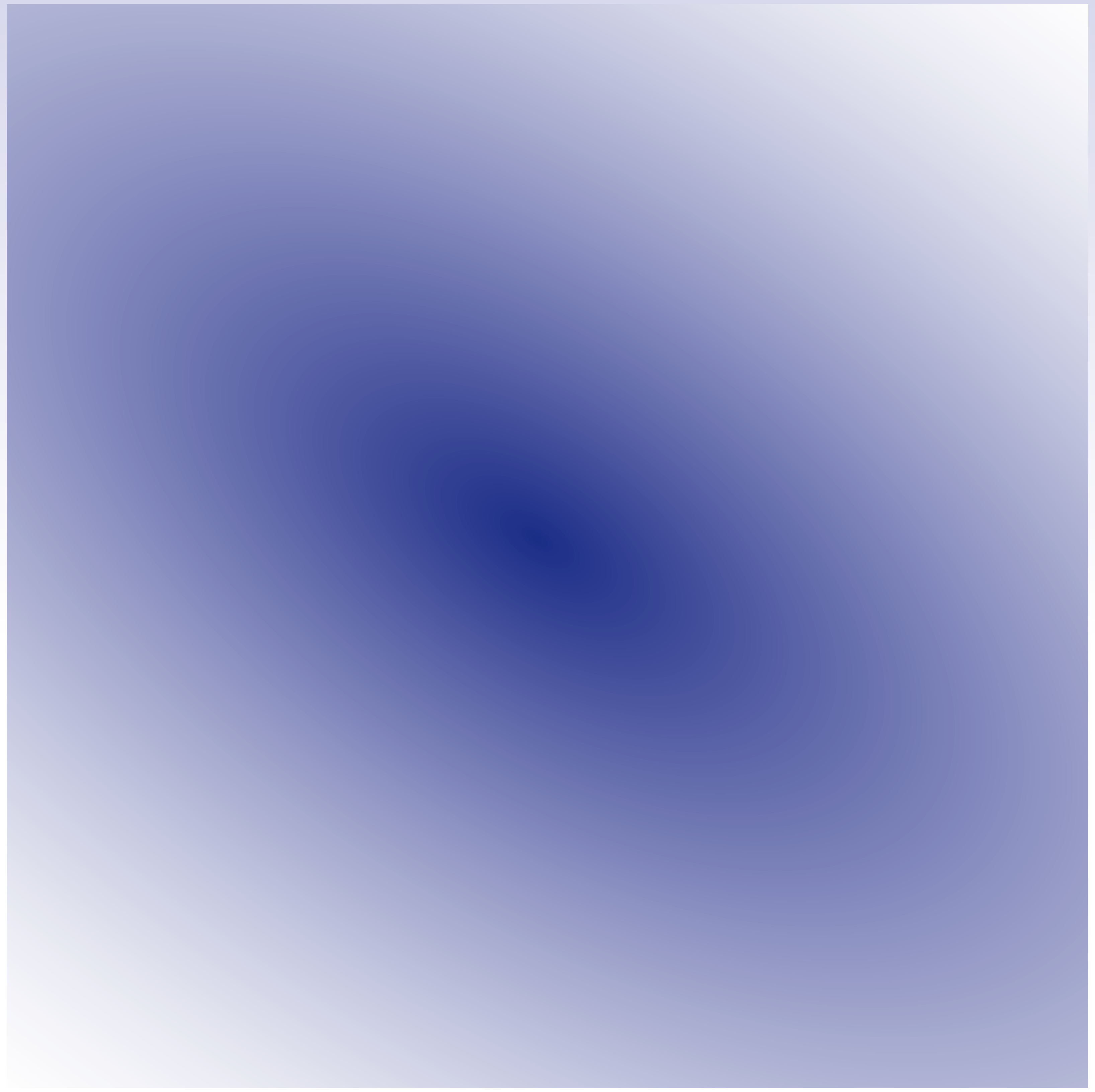
$$\nabla \times X = (\star dX^\flat)^\sharp$$

$$d \circ d = 0$$

$$d \circ d = 0$$

Analogy: *curl of gradient*

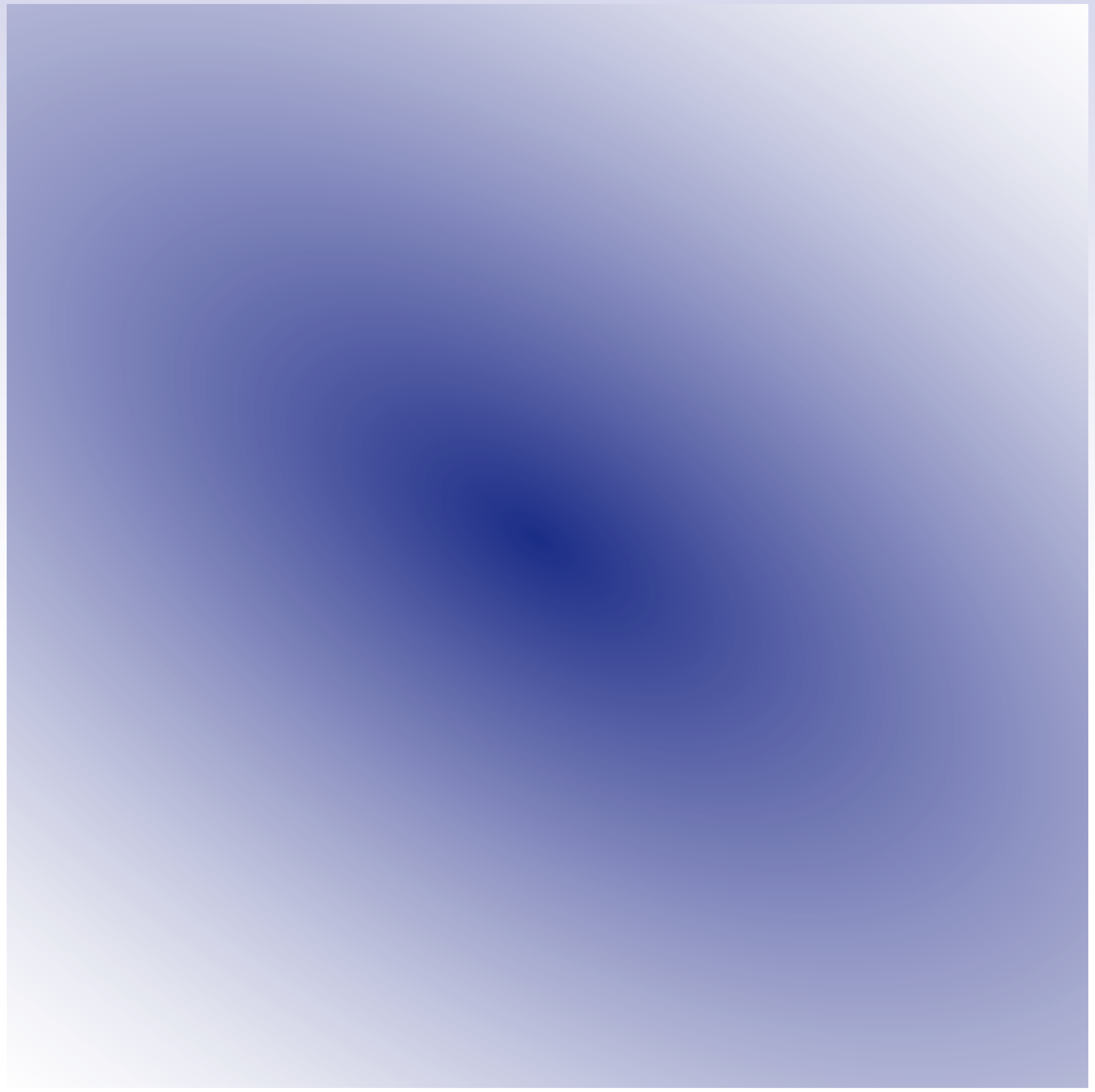
$$d \circ d = 0$$



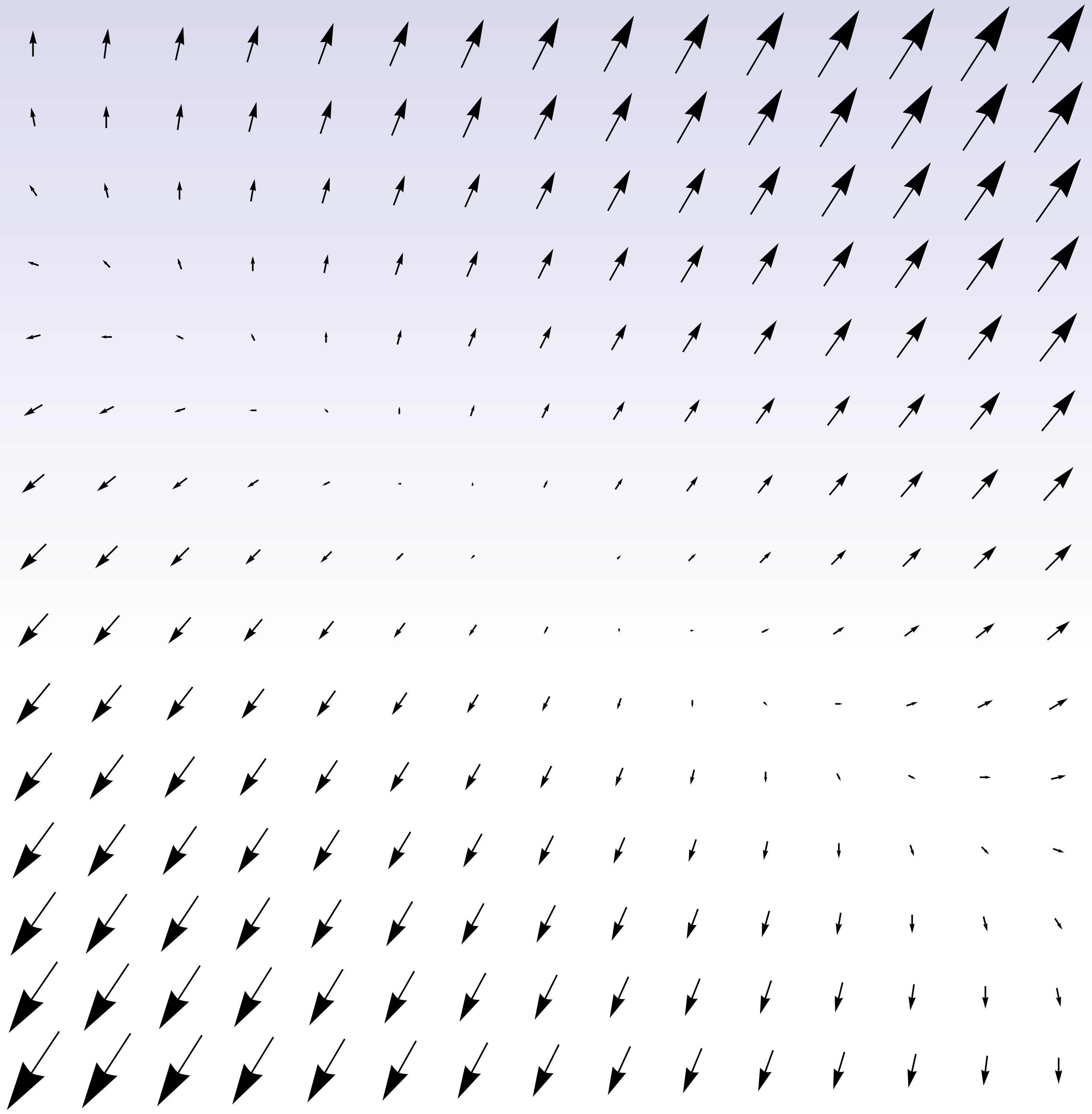
ϕ

Analogy: *curl of gradient*

$d \circ d = 0$



ϕ



grad ϕ

Analogy: *curl of gradient*

$d \circ d = 0$

ϕ | grad ϕ | curl $\circ$ grad ϕ

Analogy: *curl of gradient*

$$d \circ d = 0$$

ϕ $\quad$ $d\phi$ $\quad$ $\star dd\phi$

Analogy: *curl of gradient*

Exterior Calculus - Summary

Exterior Calculus - Summary

- k-*forms* measure k-*vectors*

Exterior Calculus - Summary

- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point

Exterior Calculus - Summary

- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point
- Exterior derivative *d* used to differentiate k-forms

Exterior Calculus - Summary

- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point
- Exterior derivative *d* used to differentiate k-forms
  - 0-form: “gradient”

Exterior Calculus - Summary

- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point
- Exterior derivative *d* used to differentiate k-forms
  - 0-form: “gradient”
  - 1-form: “curl”

Exterior Calculus - Summary

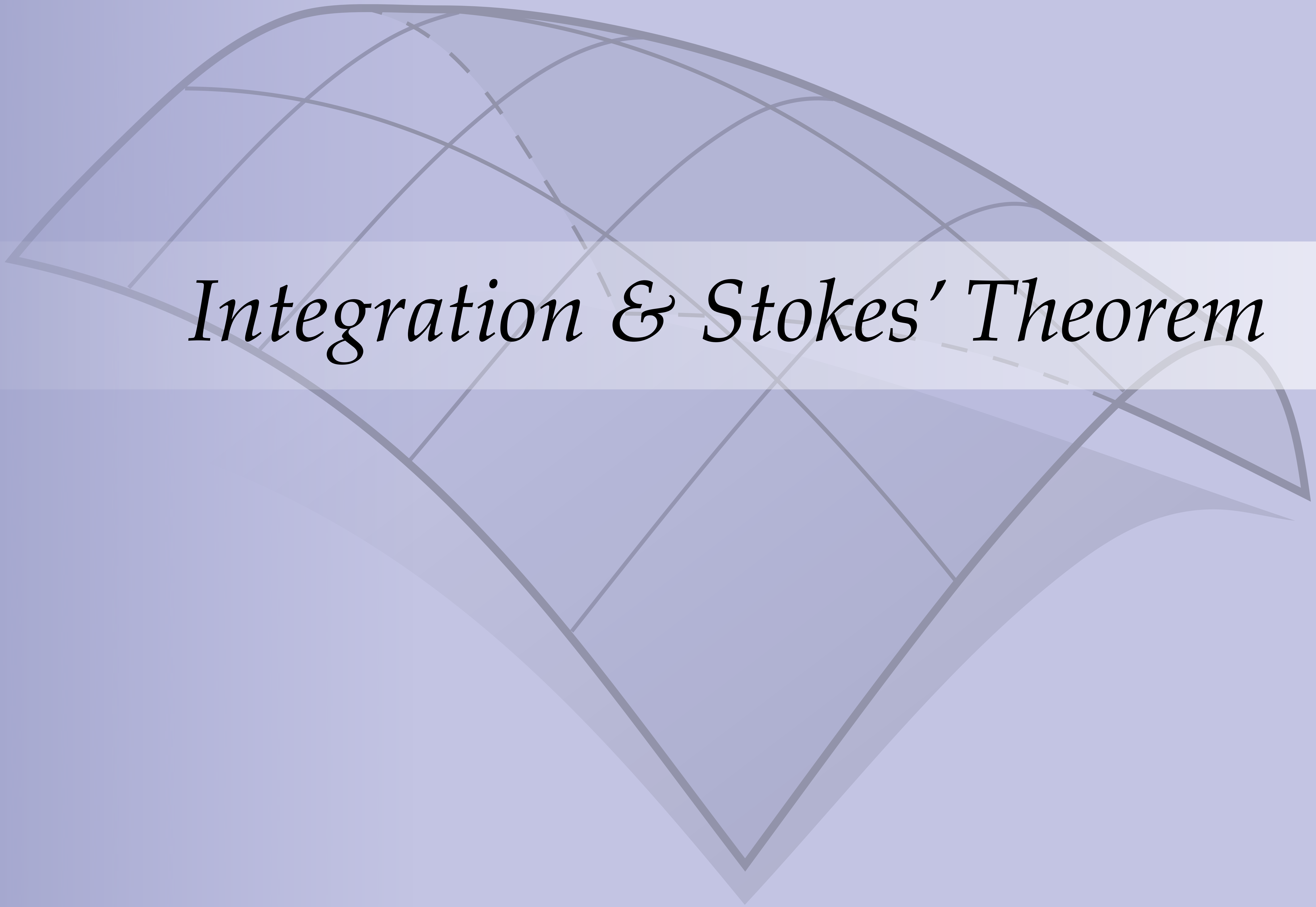
- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point
- Exterior derivative d used to differentiate k-forms
  - 0-form: “gradient”
  - 1-form: “curl”
  - 2-form: “divergence” (codifferential δ)

Exterior Calculus - Summary

- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point
- Exterior derivative *d* used to differentiate k-forms
  - 0-form: “gradient”
  - 1-form: “curl”
  - 2-form: “divergence” (codifferential δ)
  - and more...

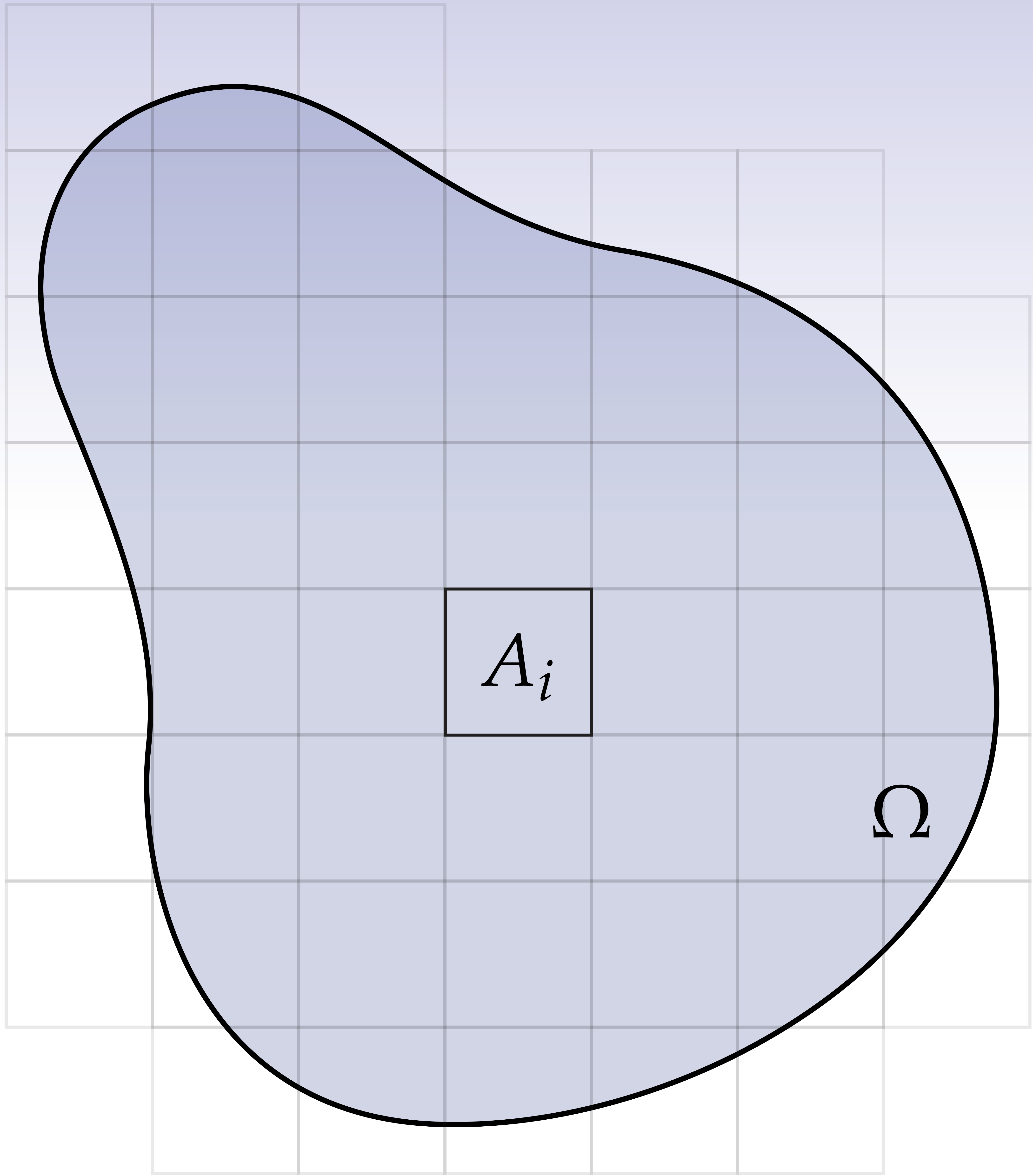
Exterior Calculus - Summary

- k-*forms* measure k-*vectors*
- *Differential* k-form is defined at each point
- Exterior derivative *d* used to differentiate k-forms
  - 0-form: “gradient”
  - 1-form: “curl”
  - 2-form: “divergence” (codifferential δ)
  - and more...
- ***d* of *d* is zero**



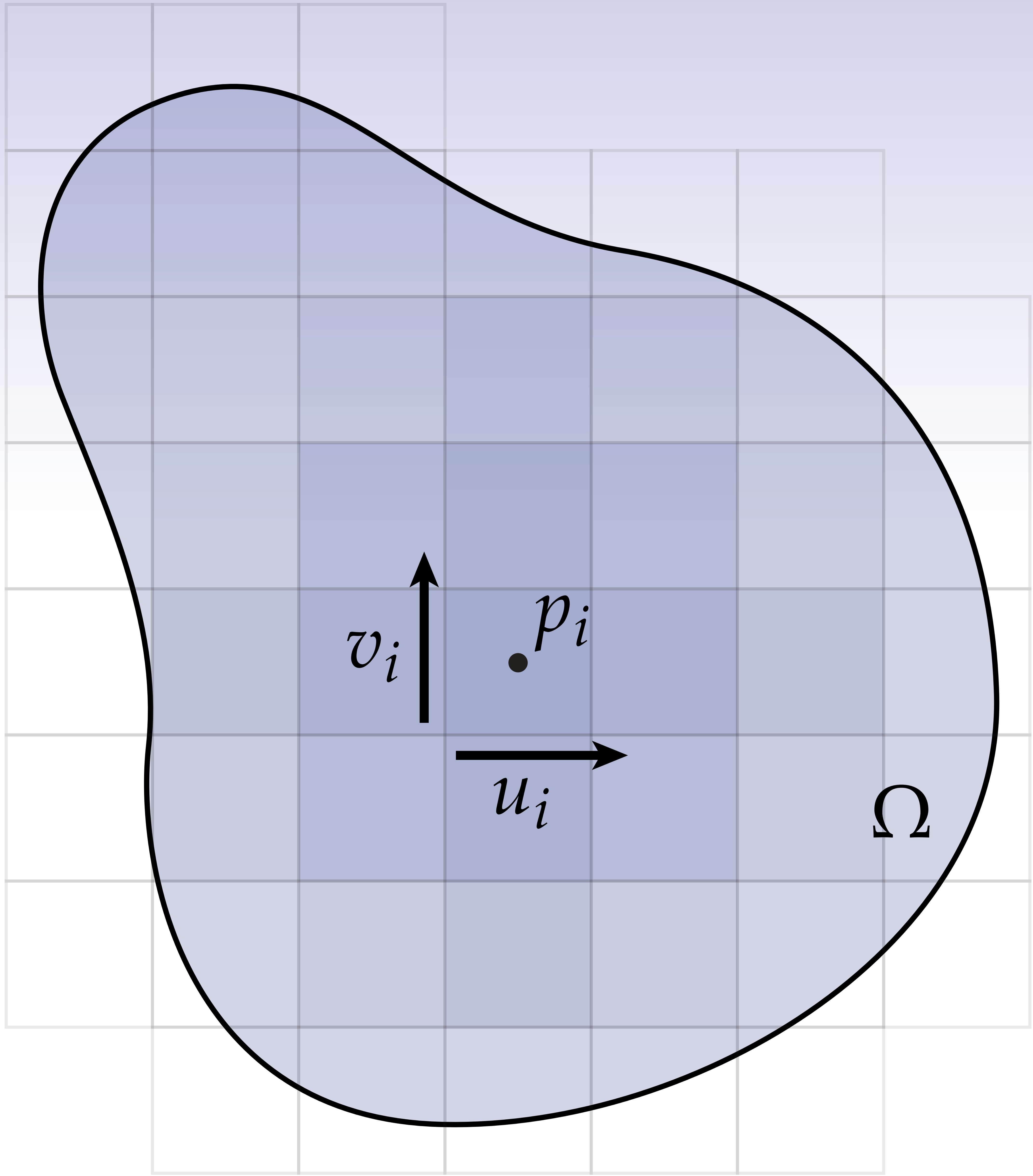
Integration & Stokes' Theorem

Area



$$\sum_i A_i \implies \int_\Omega dA$$

Integration

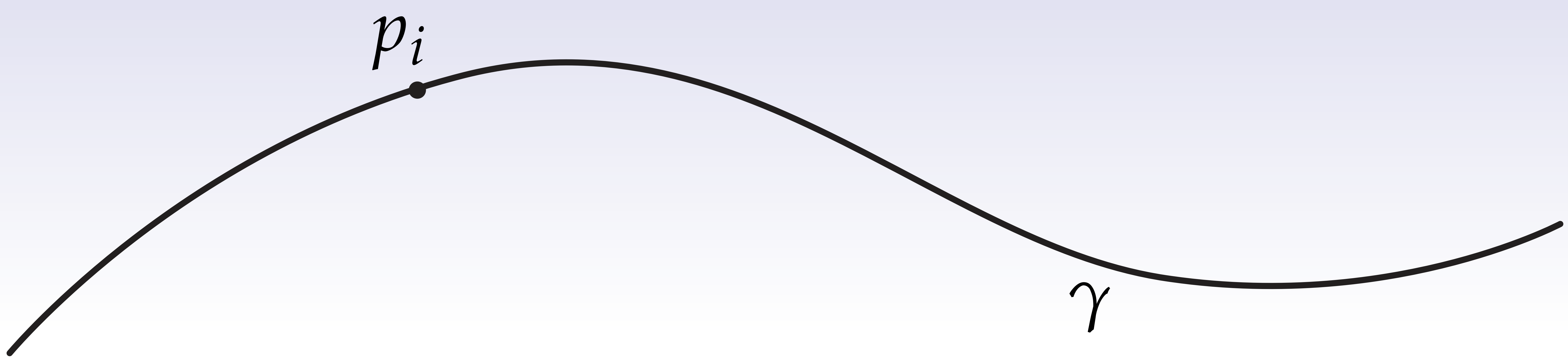


$$\sum_i \omega_{p_i}(u_i, v_i) \implies \int_\Omega \omega$$

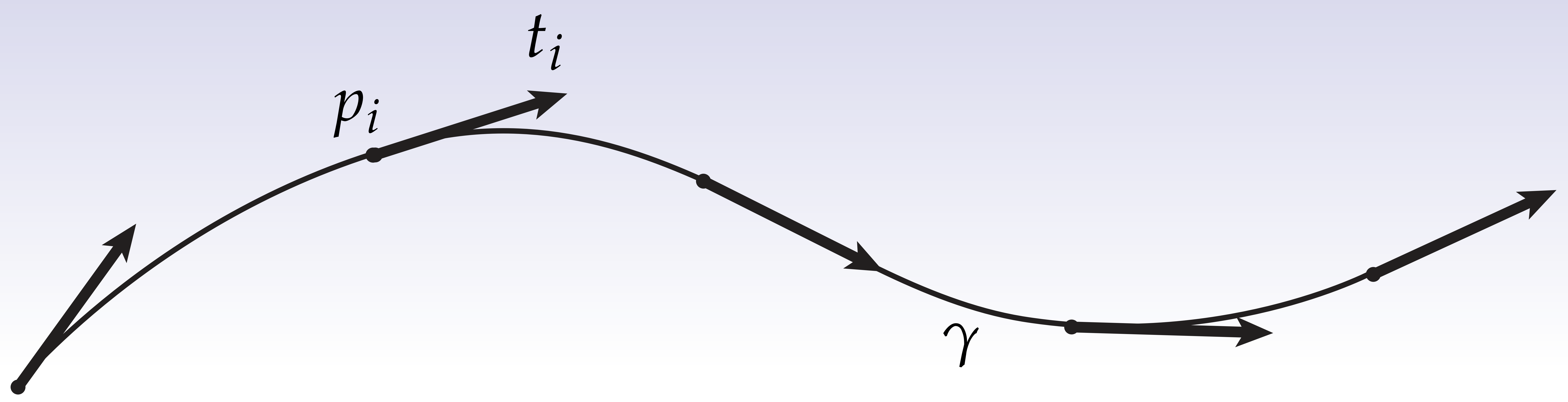
Integration on Curves

Integration on Curves

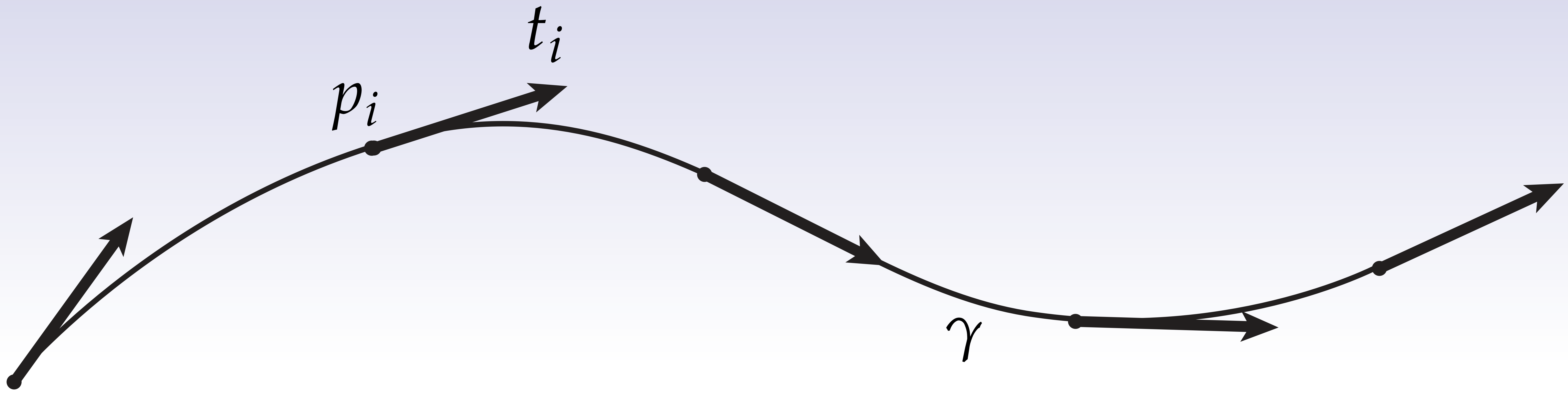
Integration on Curves



Integration on Curves

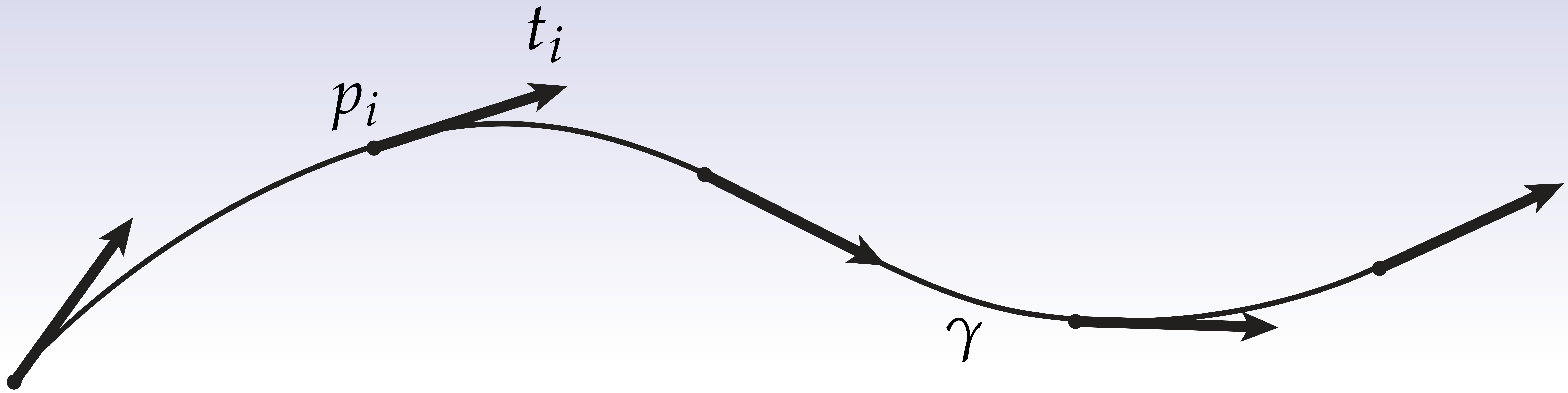


Integration on Curves

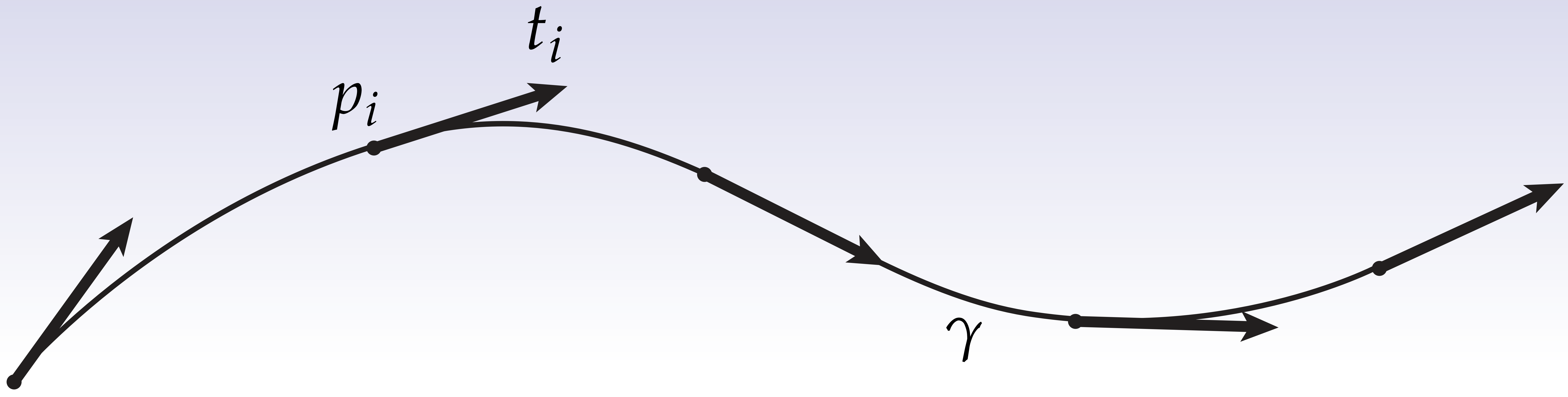


$$\int_\gamma \alpha \approx \sum_i \alpha_{p_i}(t_i)$$

Integration on Curves

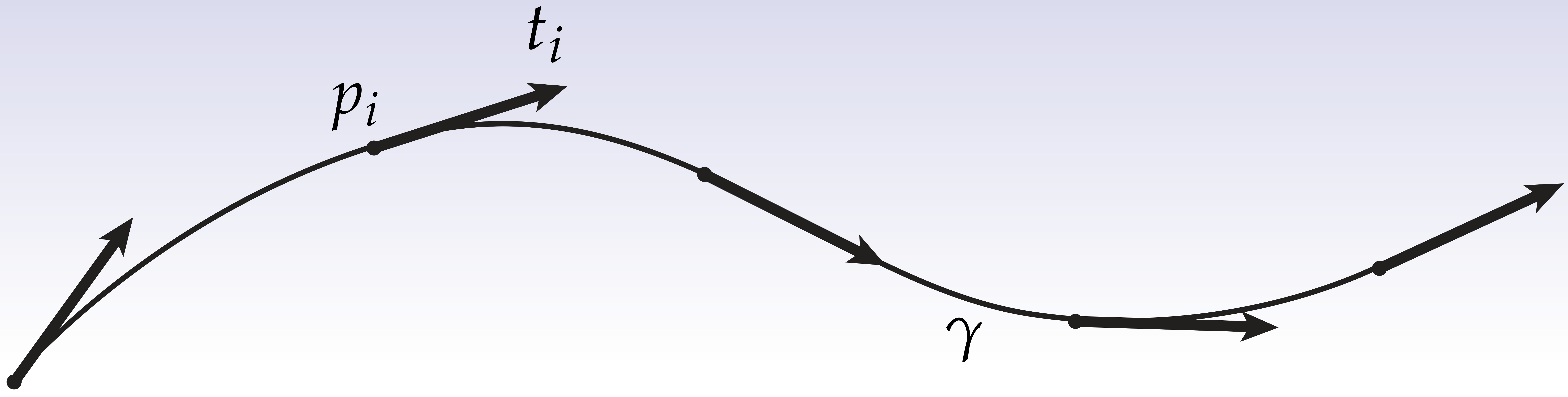


Integration on Curves



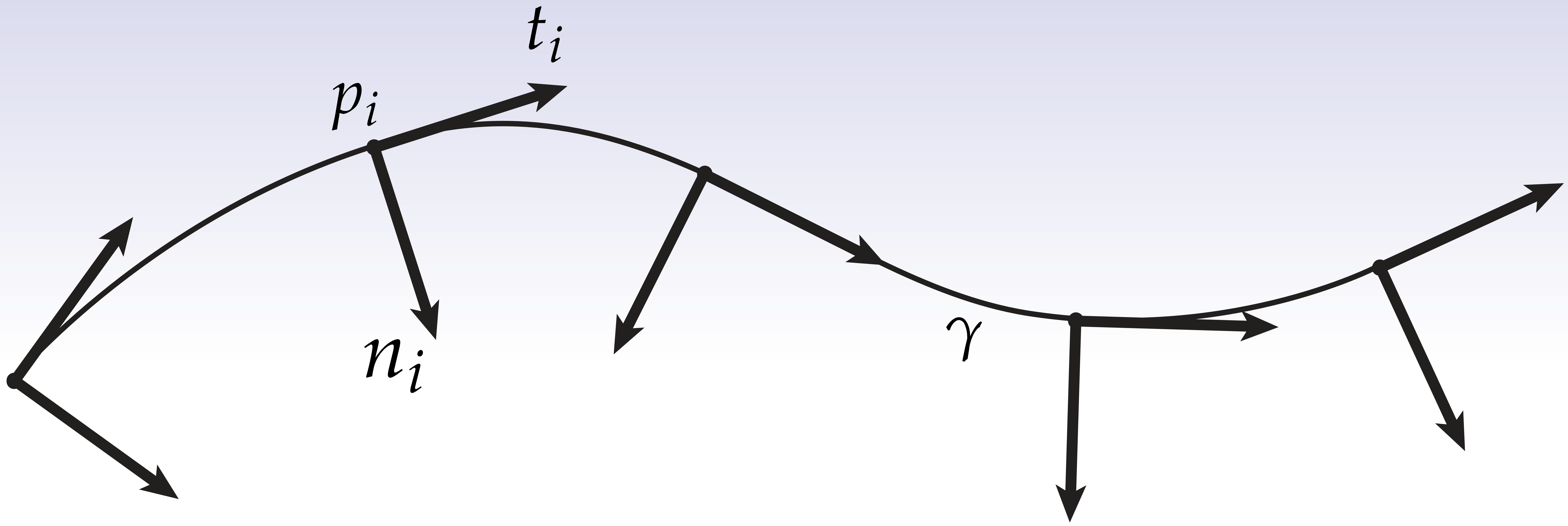
$$\int_\gamma \star\alpha$$

Integration on Curves



$$\int_\gamma \star\alpha \approx \sum_i \star\alpha_{p_i}(t_i)$$

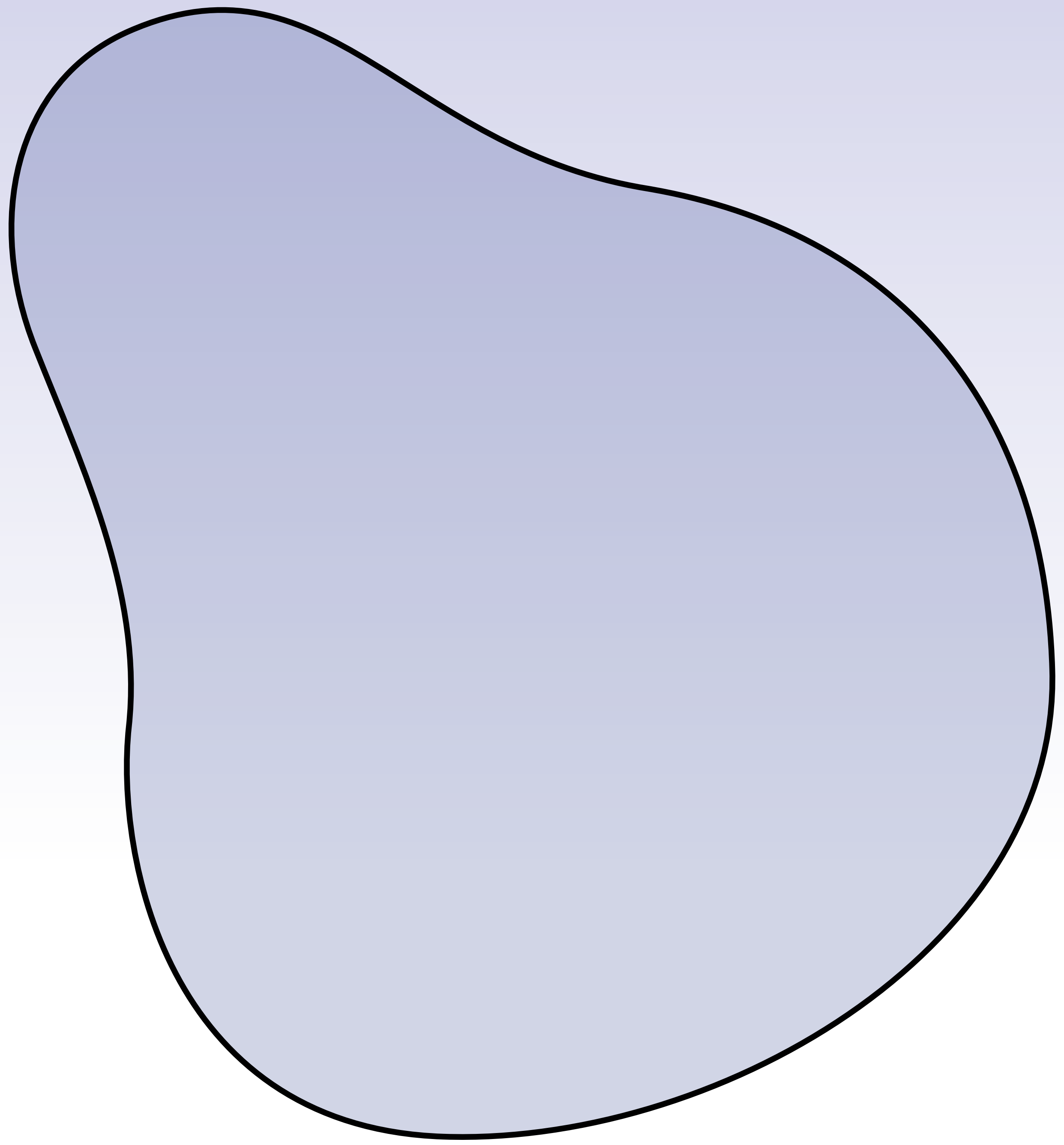
Integration on Curves



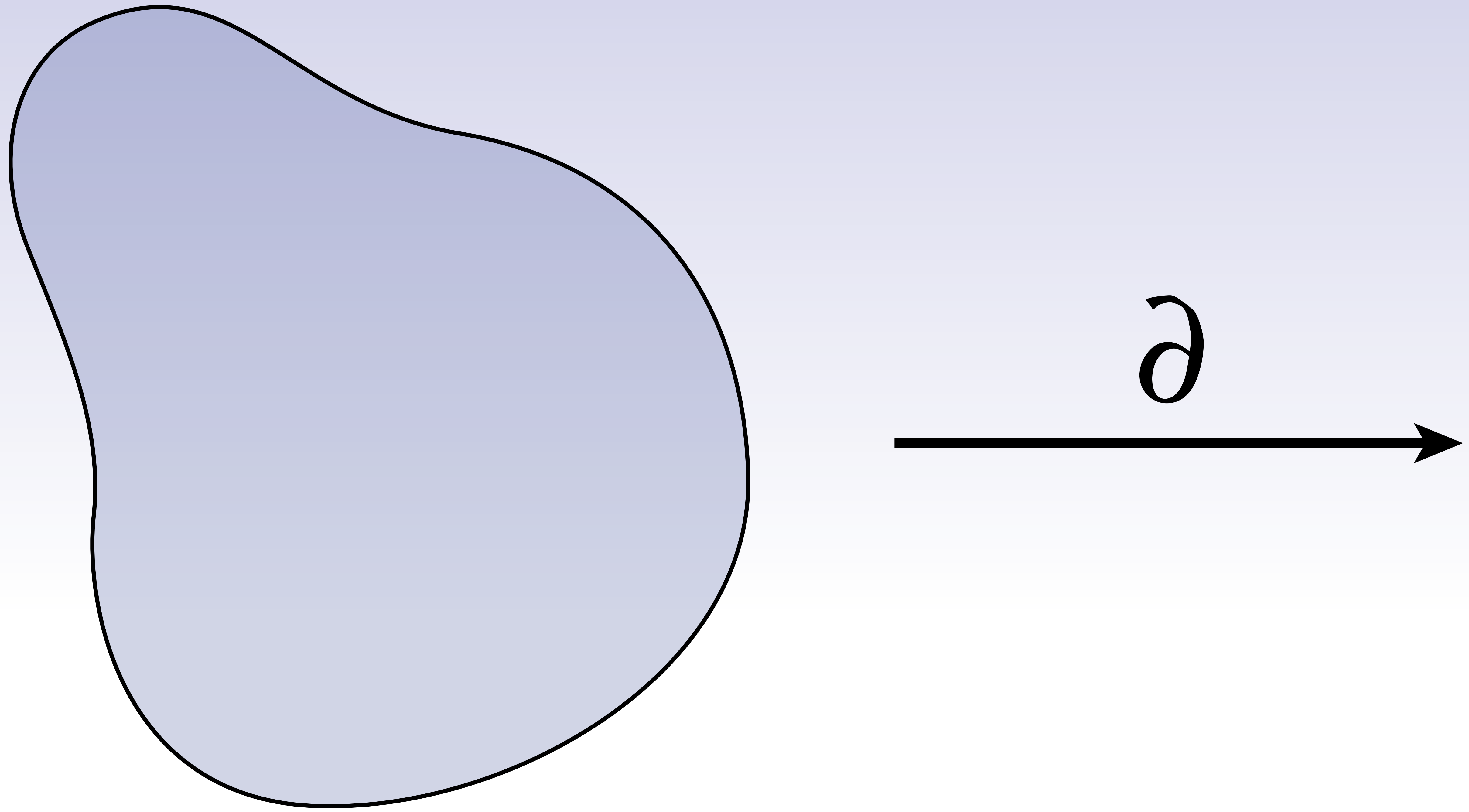
$$\int_\gamma \star\alpha \approx \sum_i \star\alpha_{p_i}(t_i) = \sum_i \alpha_{p_i}(n_i)$$

Boundary

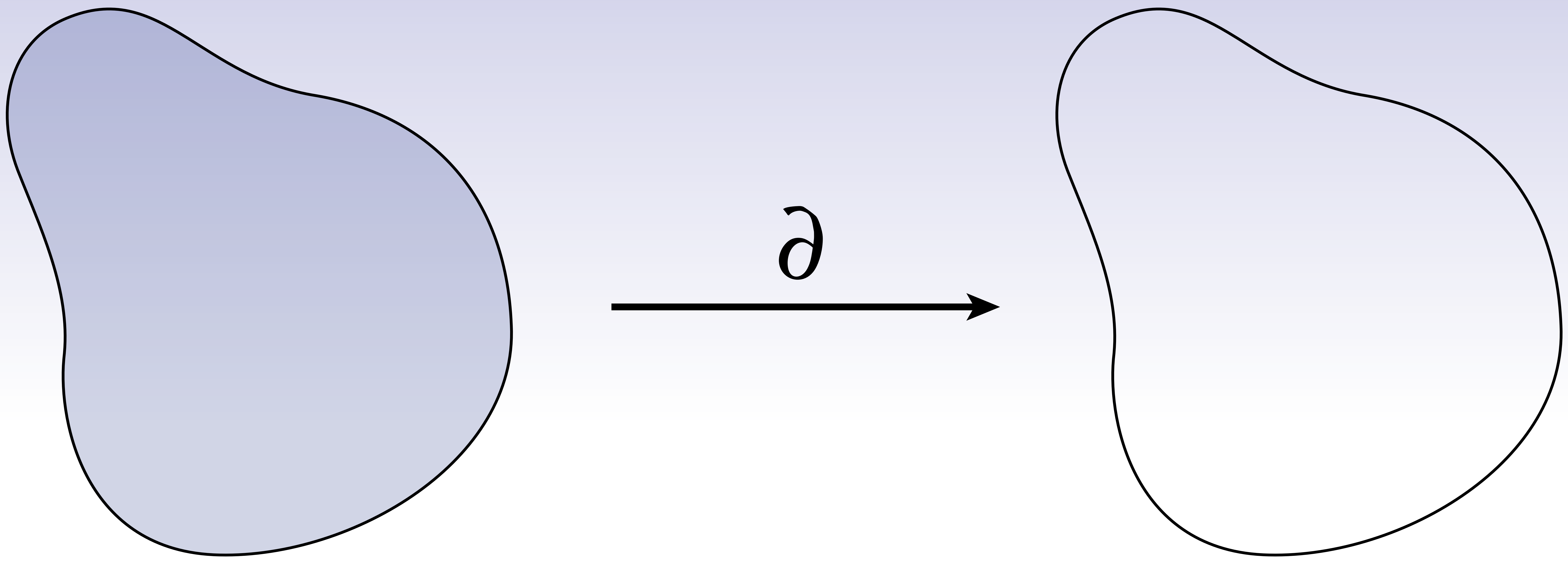
Boundary



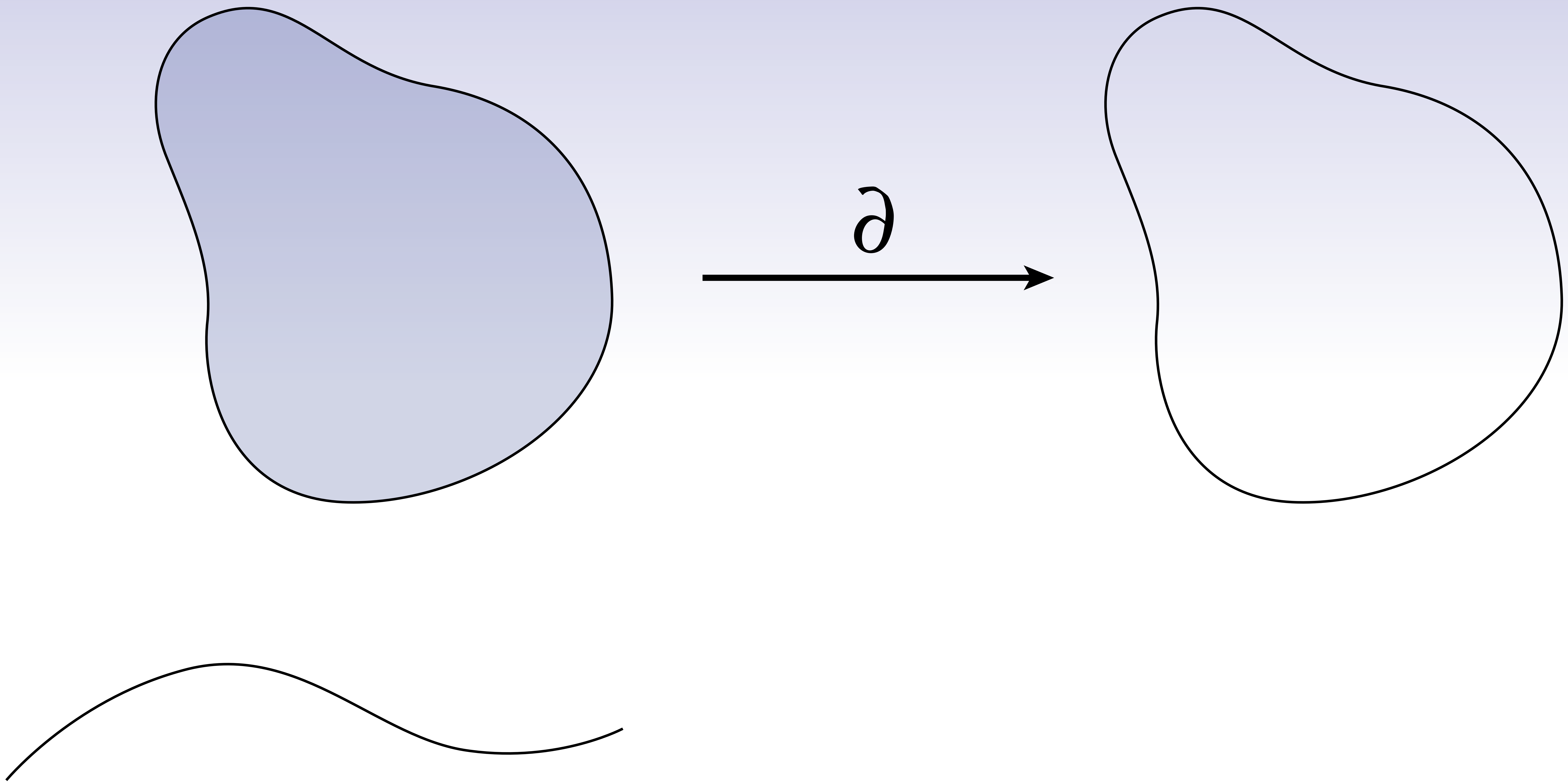
Boundary



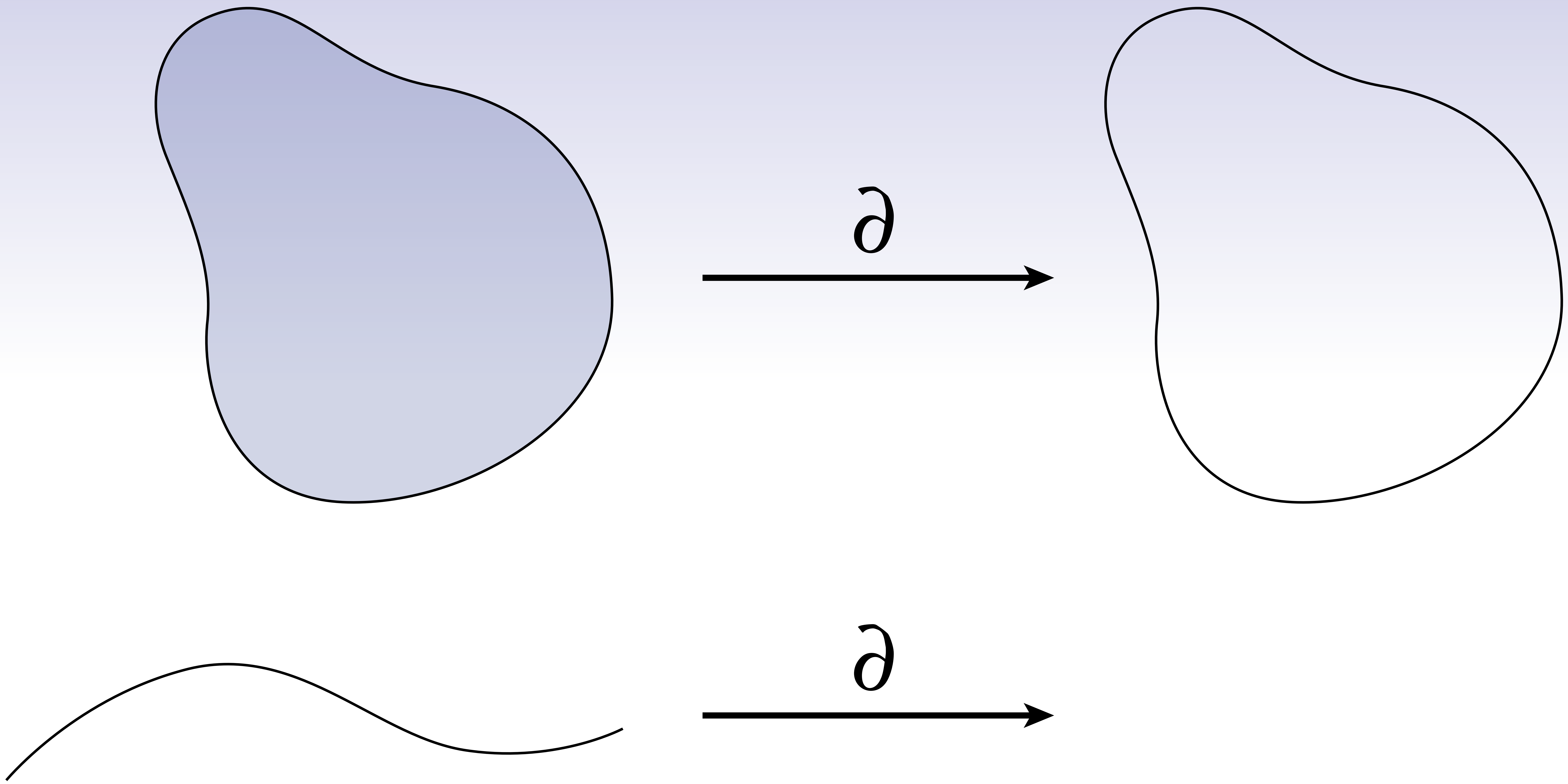
Boundary



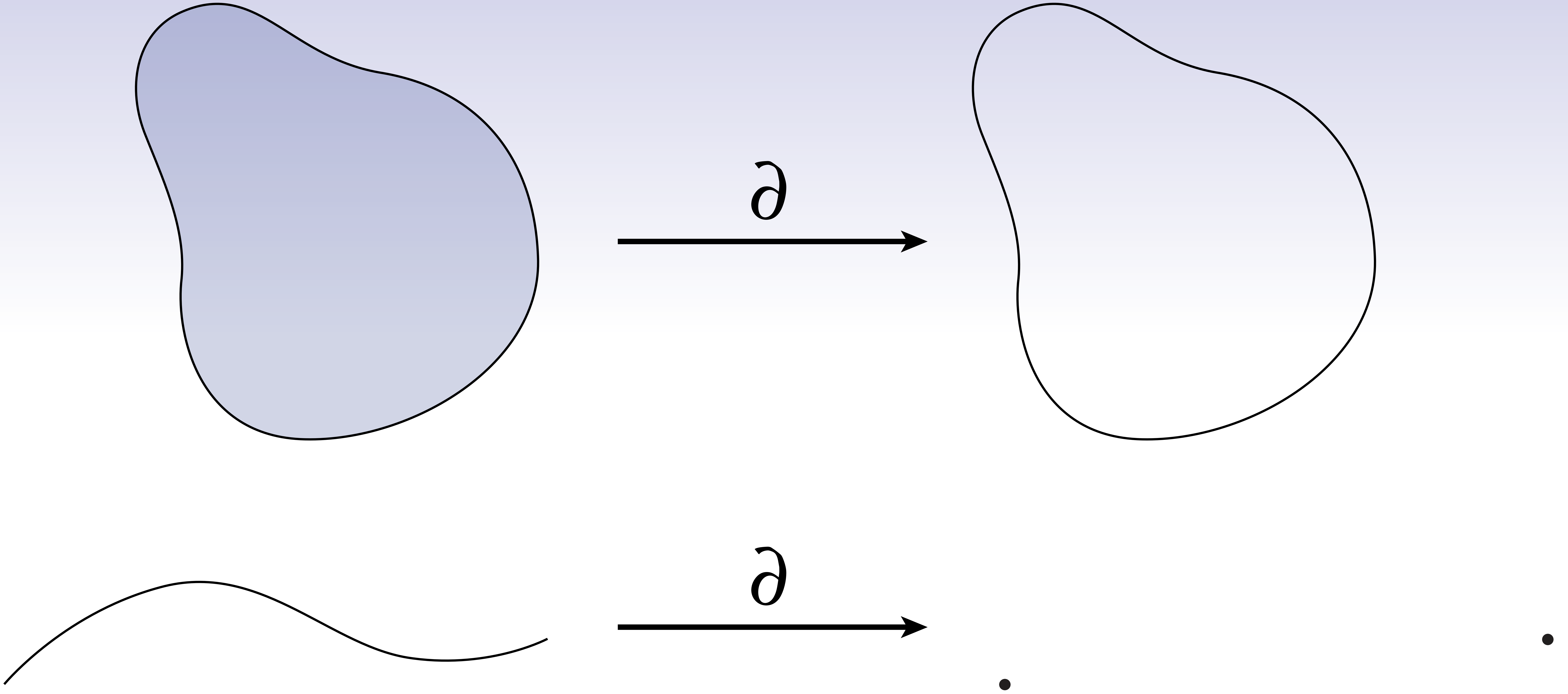
Boundary



Boundary



Boundary

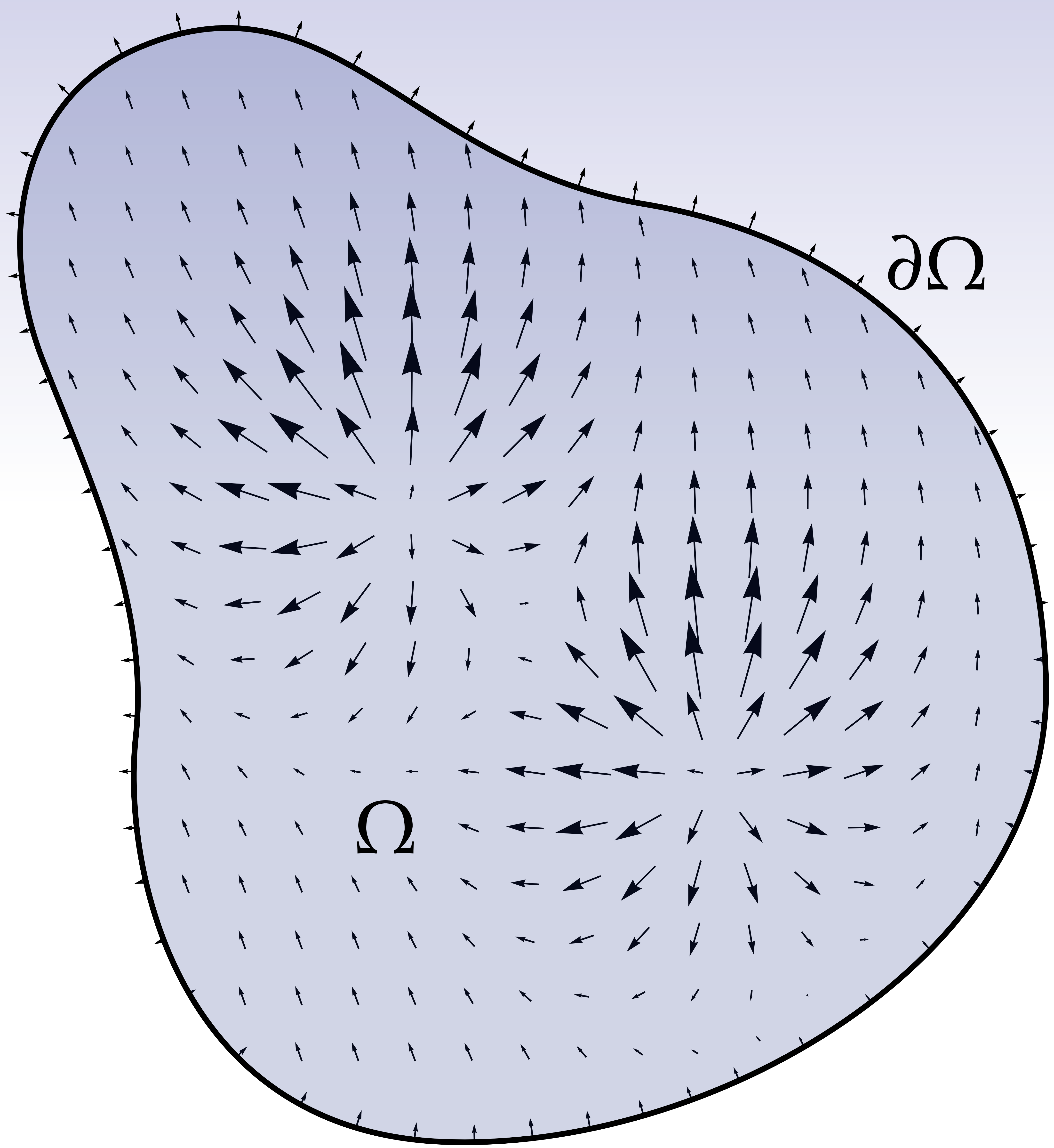


Stokes' Theorem

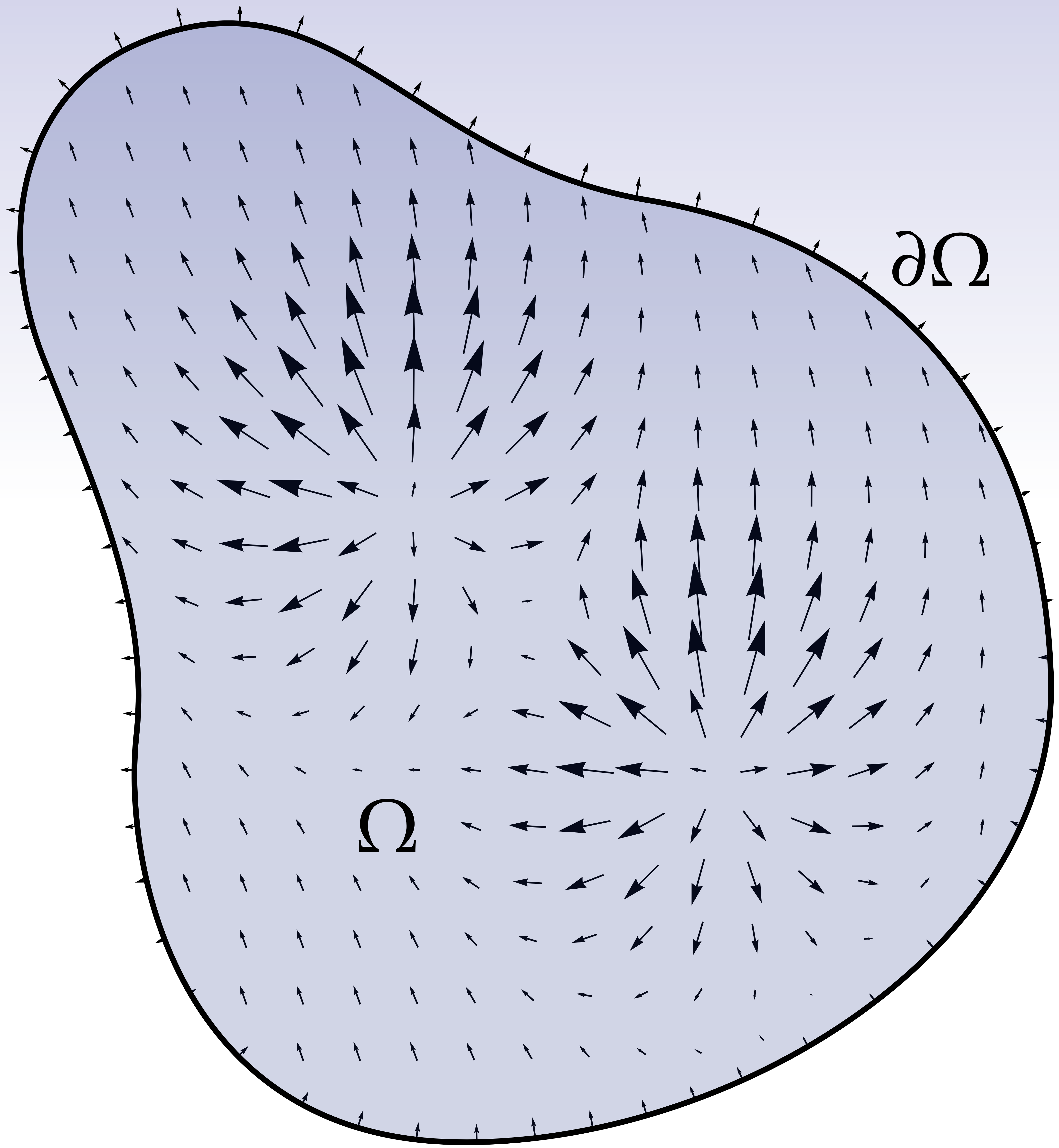
$$\int_{\Omega} d\alpha = \int_{\partial\Omega} \alpha$$

Example: Divergence Theorem

Example: Divergence Theorem

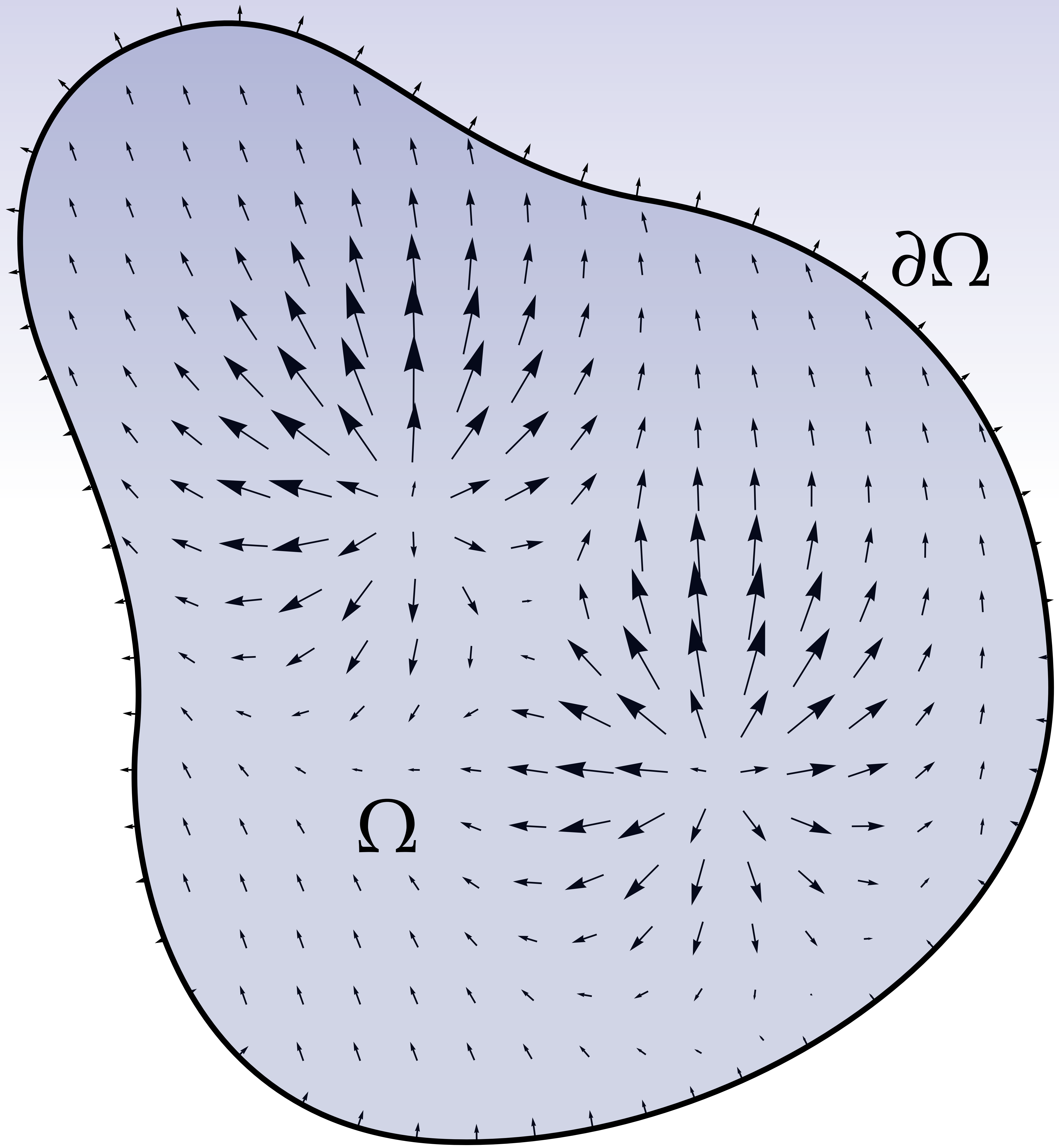


Example: Divergence Theorem



$$\int_\Omega \nabla \cdot X \, dA = \int_{\partial\Omega} n \cdot X \, d\ell$$

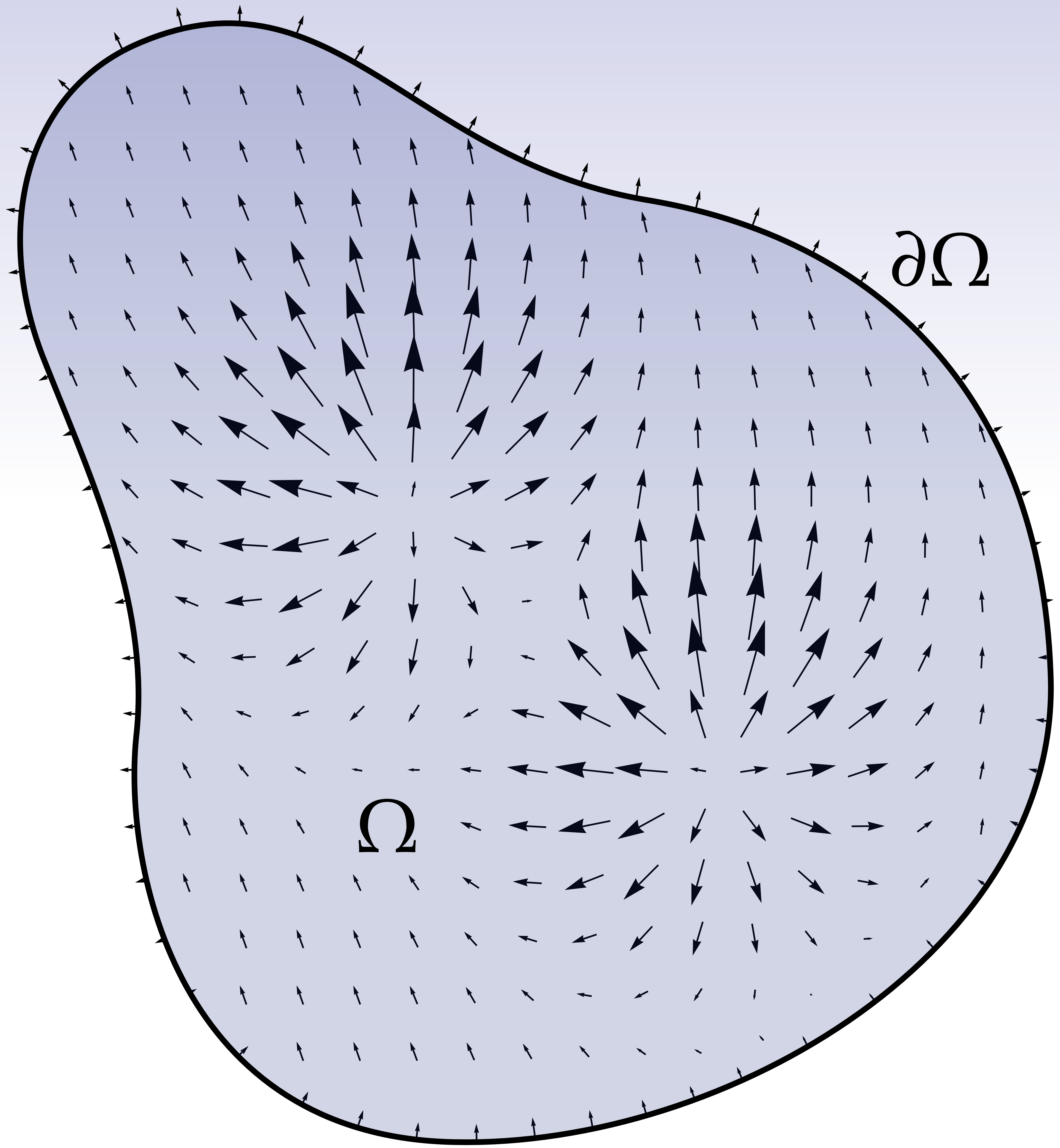
Example: Divergence Theorem



$$\int_\Omega \nabla \cdot X \, dA = \int_{\partial\Omega} n \cdot X \, d\ell$$

What goes in, must come out!

Example: Divergence Theorem

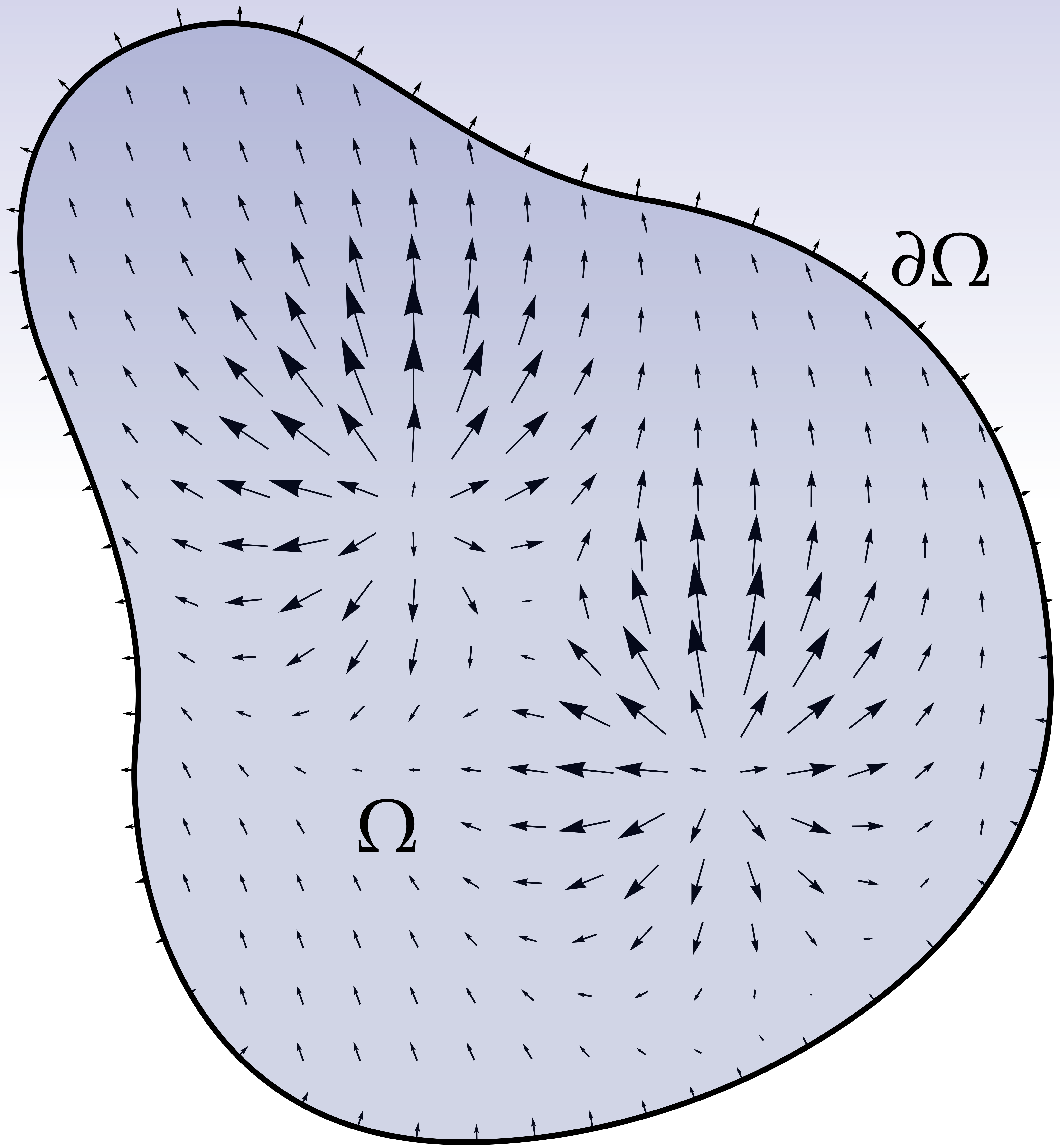


$$\int_{\Omega} \nabla \cdot X \, dA = \int_{\partial\Omega} n \cdot X \, d\ell$$

$$\Longleftrightarrow$$

What goes in, must come out!

Example: Divergence Theorem



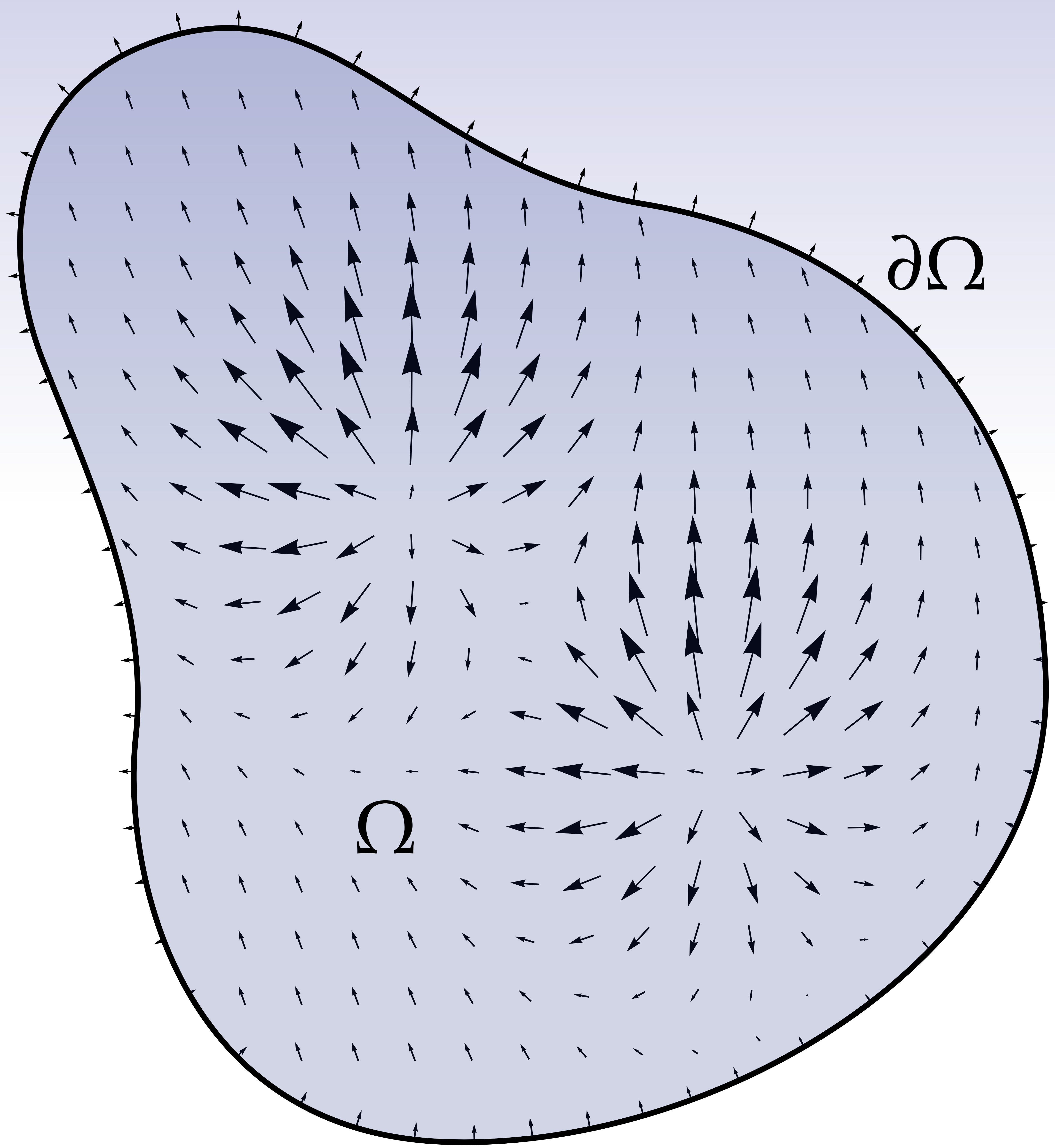
$$\int_\Omega \nabla \cdot X \, dA = \int_{\partial\Omega} n \cdot X \, d\ell$$

$$\Longleftrightarrow$$

$$\int_\Omega d \star X^\flat$$

What goes in, must come out!

Example: Divergence Theorem



$$\int_\Omega \nabla \cdot X \, dA = \int_{\partial\Omega} n \cdot X \, d\ell$$

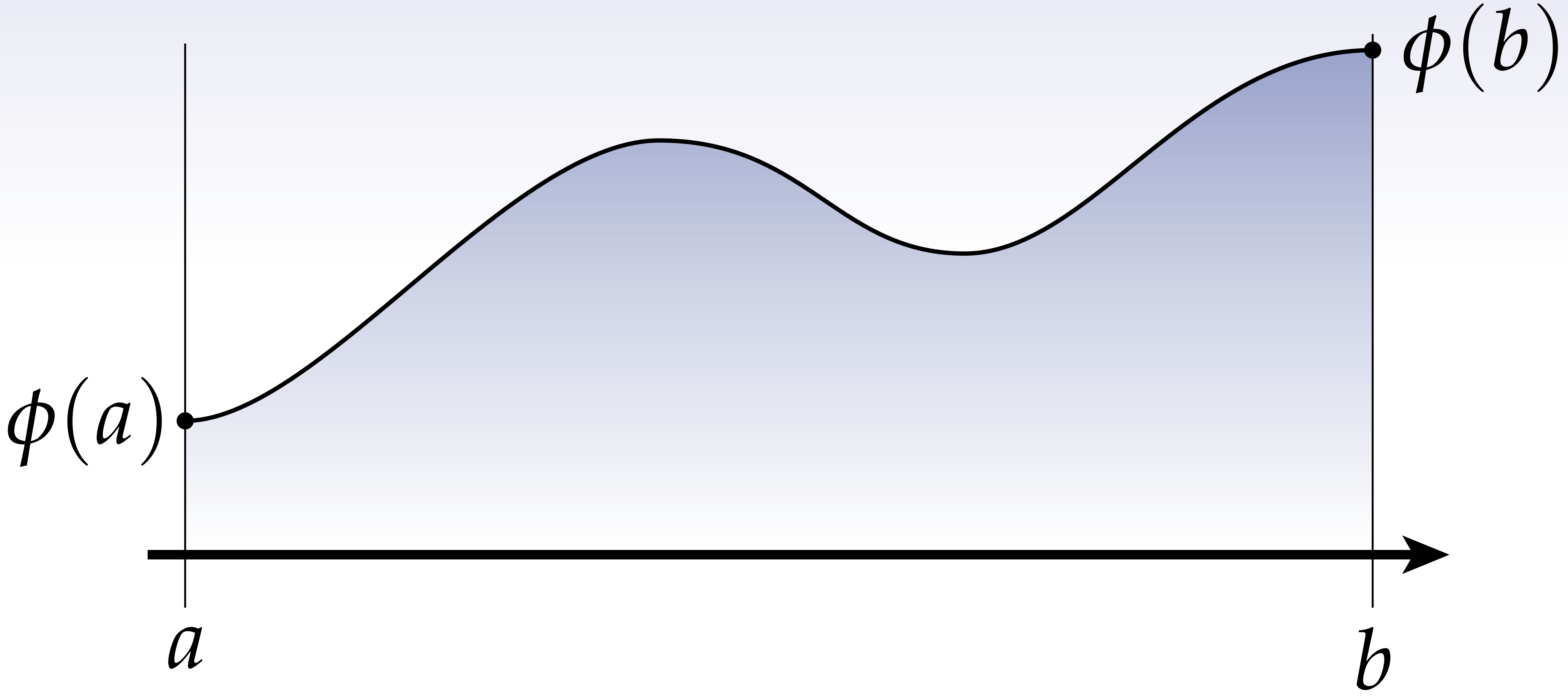
$$\Longleftrightarrow$$

$$\int_\Omega d \star X^\flat = \int_{\partial\Omega} \star X^\flat$$

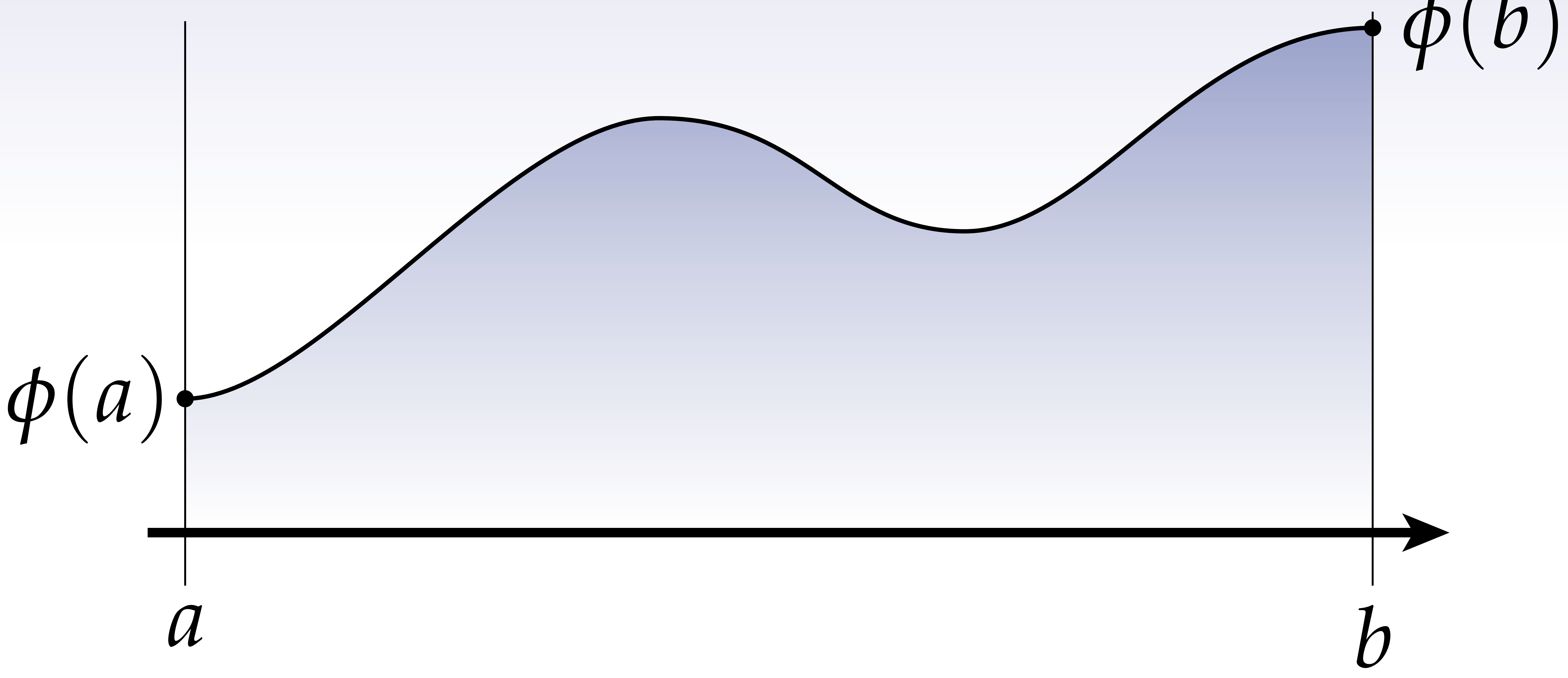
What goes in, must come out!

Example: Fundamental Theorem of Calculus

Example: Fundamental Theorem of Calculus

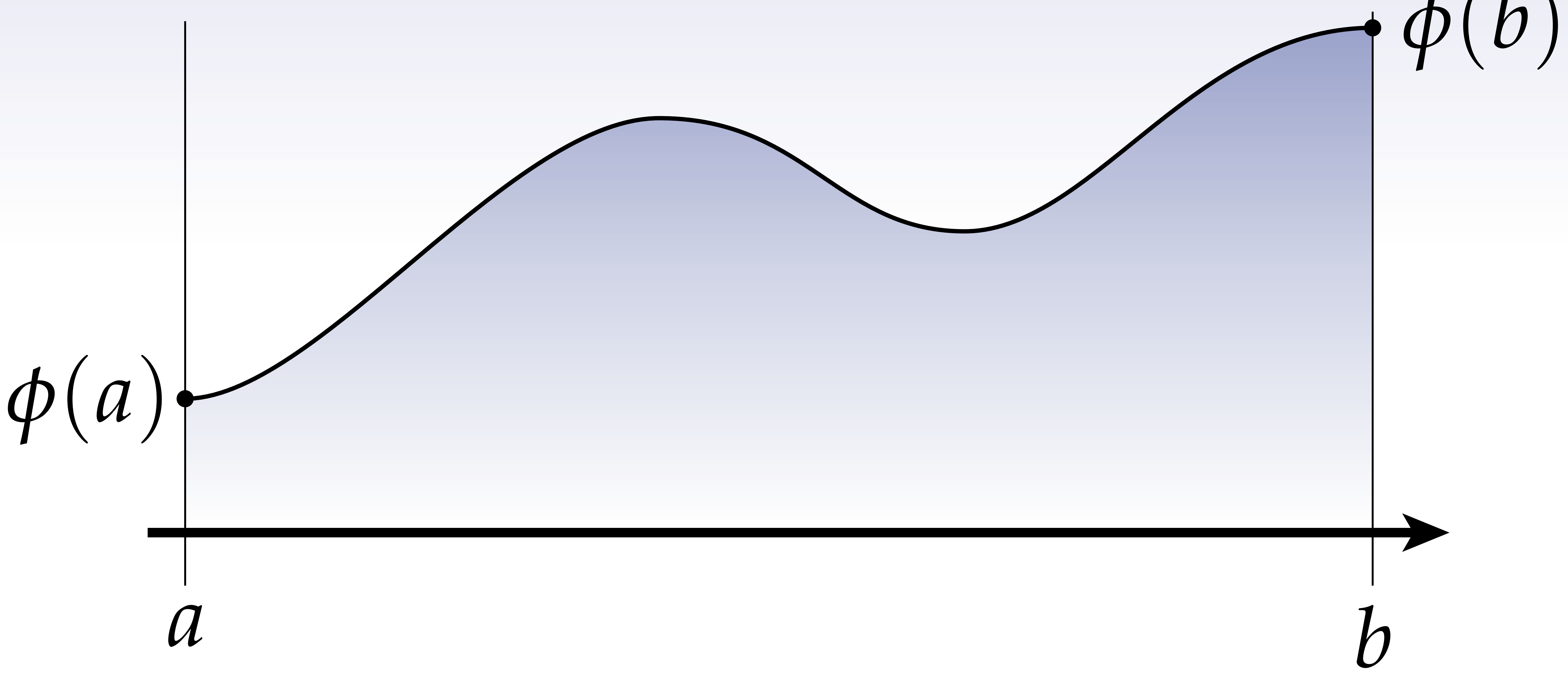


Example: Fundamental Theorem of Calculus



$$\int_a^b \frac{\partial \phi}{\partial x} dx = \phi(b) - \phi(a)$$

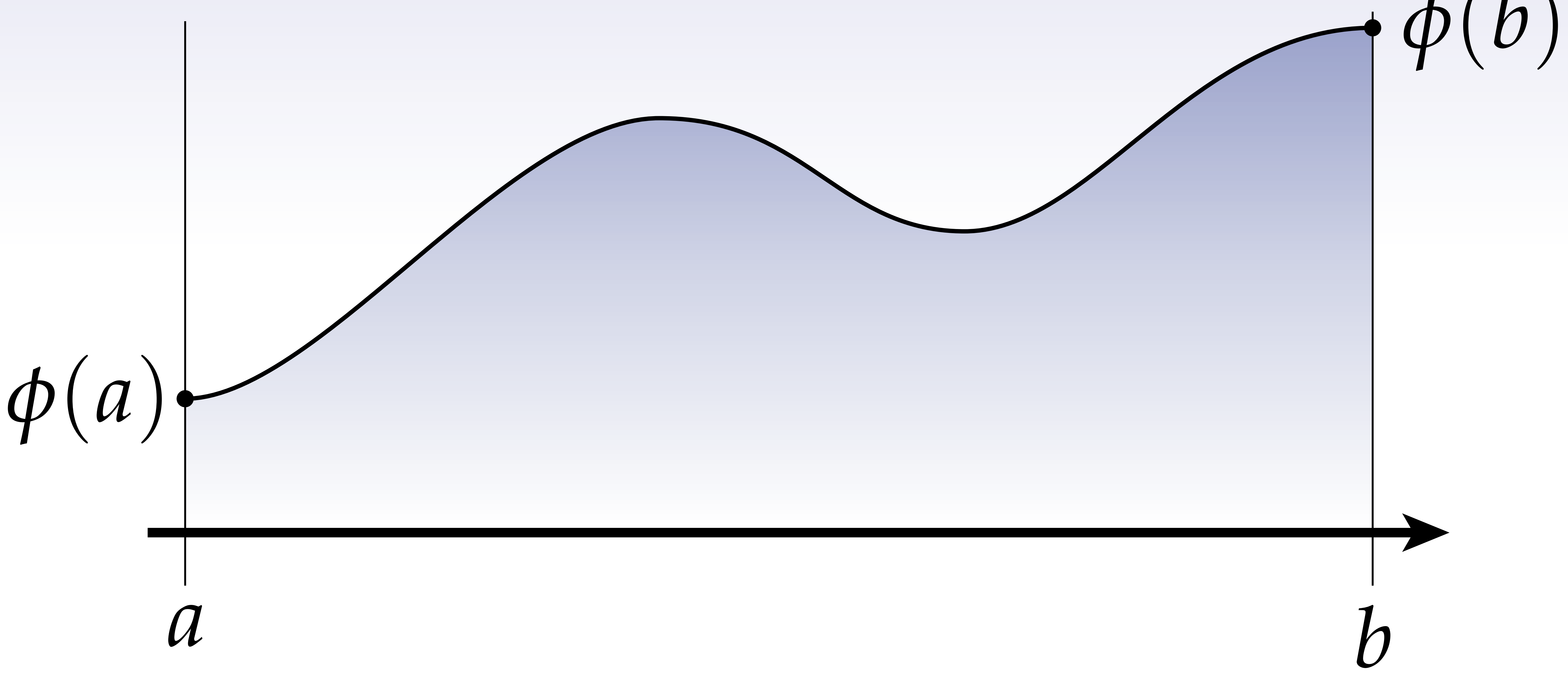
Example: Fundamental Theorem of Calculus



$$\int_a^b \frac{\partial \phi}{\partial x} dx = \phi(b) - \phi(a)$$

$$\Longleftrightarrow$$

Example: Fundamental Theorem of Calculus

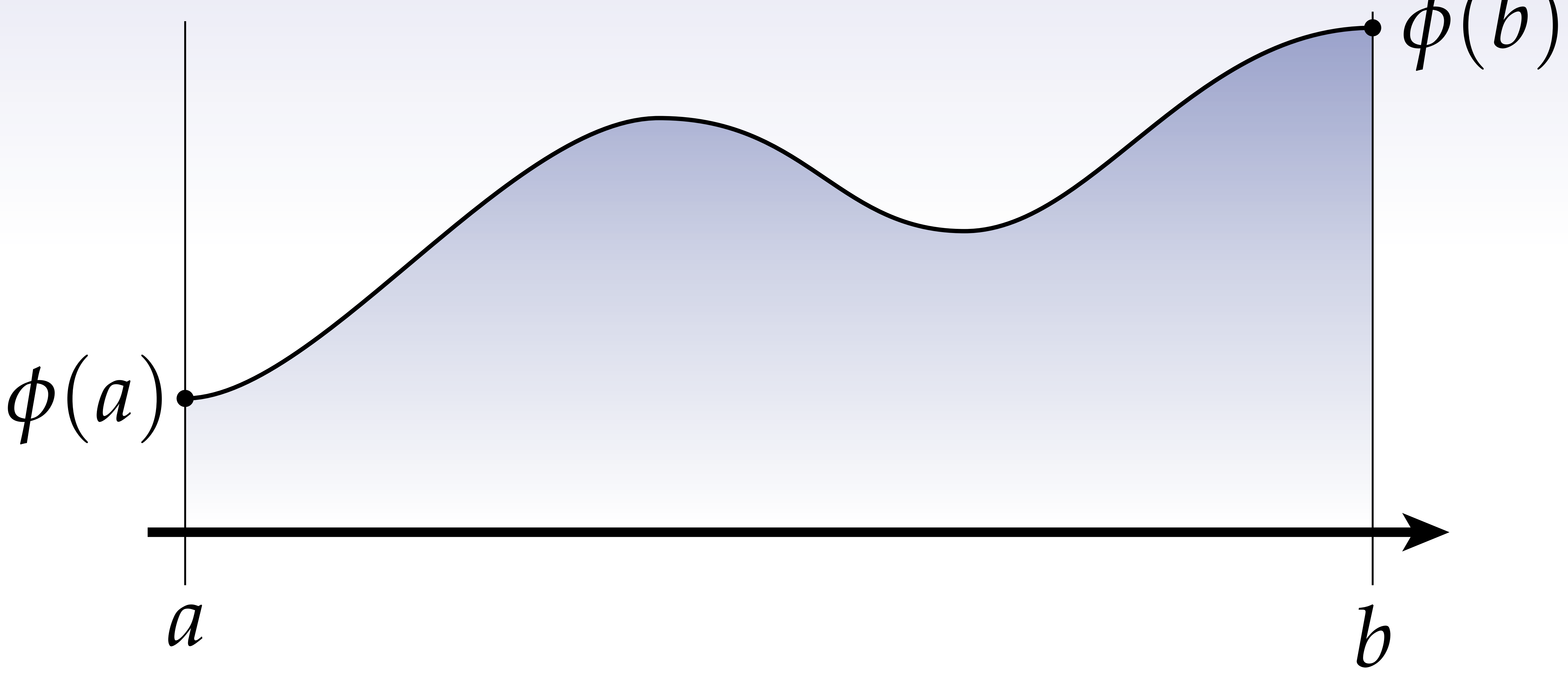


$$\int_a^b \frac{\partial \phi}{\partial x} dx = \phi(b) - \phi(a)$$

$$\Longleftrightarrow$$

$$\int_{[a,b]} d\phi$$

Example: Fundamental Theorem of Calculus



$$\int_a^b \frac{\partial \phi}{\partial x} dx = \phi(b) - \phi(a)$$

$$\Longleftrightarrow$$

$$\int_{[a,b]} d\phi = \int_{\partial[a,b]} \phi$$

Integration & Stokes' Theorem - Summary

Integration & Stokes' Theorem - Summary

- Integration

Integration & Stokes' Theorem - Summary

- Integration
  - break domain into small pieces

Integration & Stokes' Theorem - Summary

- Integration
  - break domain into small pieces
  - measure each piece with k-form

Integration & Stokes' Theorem - Summary

- Integration
  - break domain into small pieces
  - measure each piece with k-form
- Stokes' theorem

Integration & Stokes' Theorem - Summary

- Integration
  - break domain into small pieces
  - measure each piece with k-form
- Stokes' theorem
  - convert region integral to boundary integral

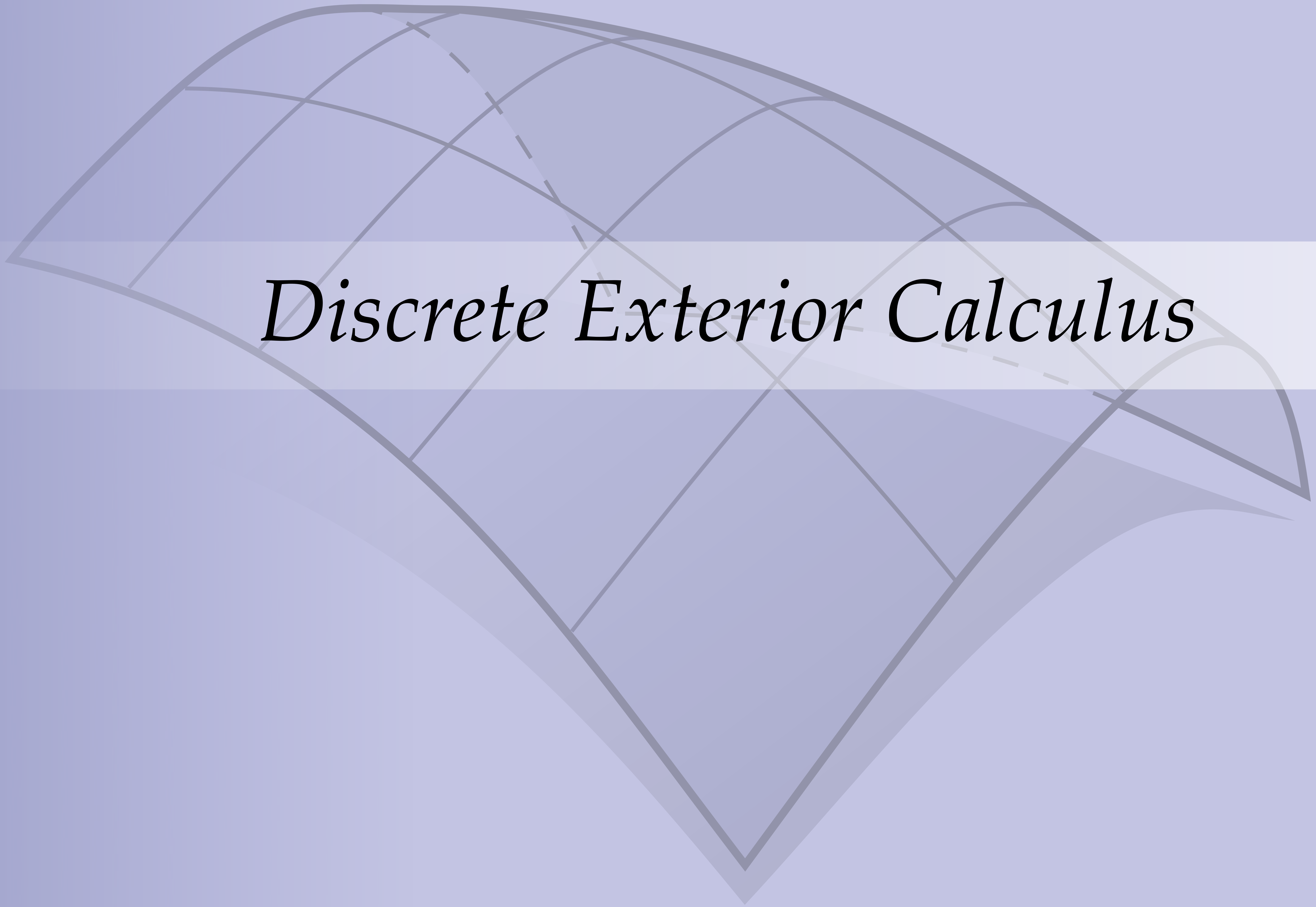
Integration & Stokes' Theorem - Summary

- Integration
  - break domain into small pieces
  - measure each piece with k-form
- Stokes' theorem
  - convert region integral to boundary integral
  - special cases: divergence theorem, F.T.C., *many more!*

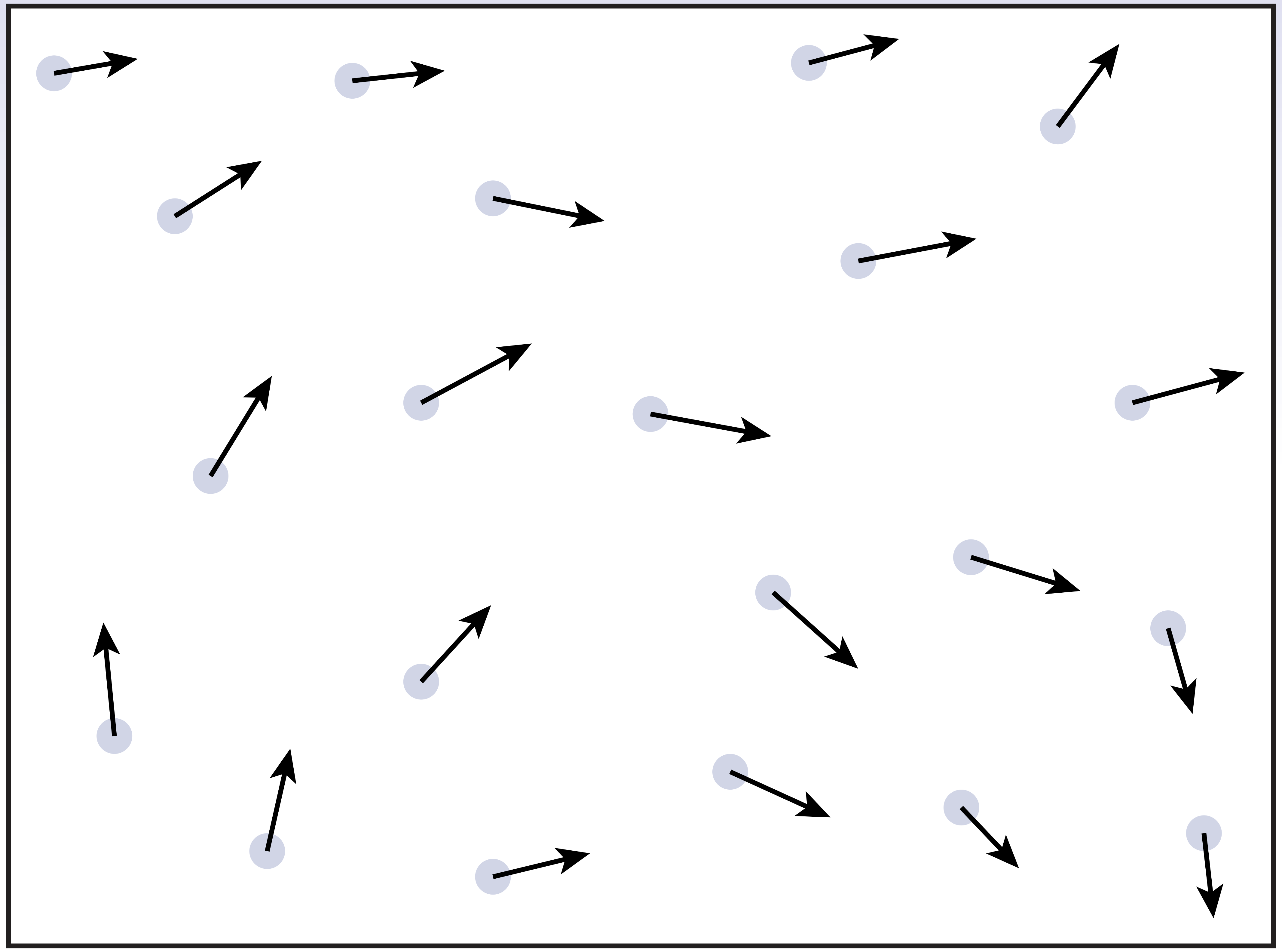
Integration & Stokes' Theorem - Summary

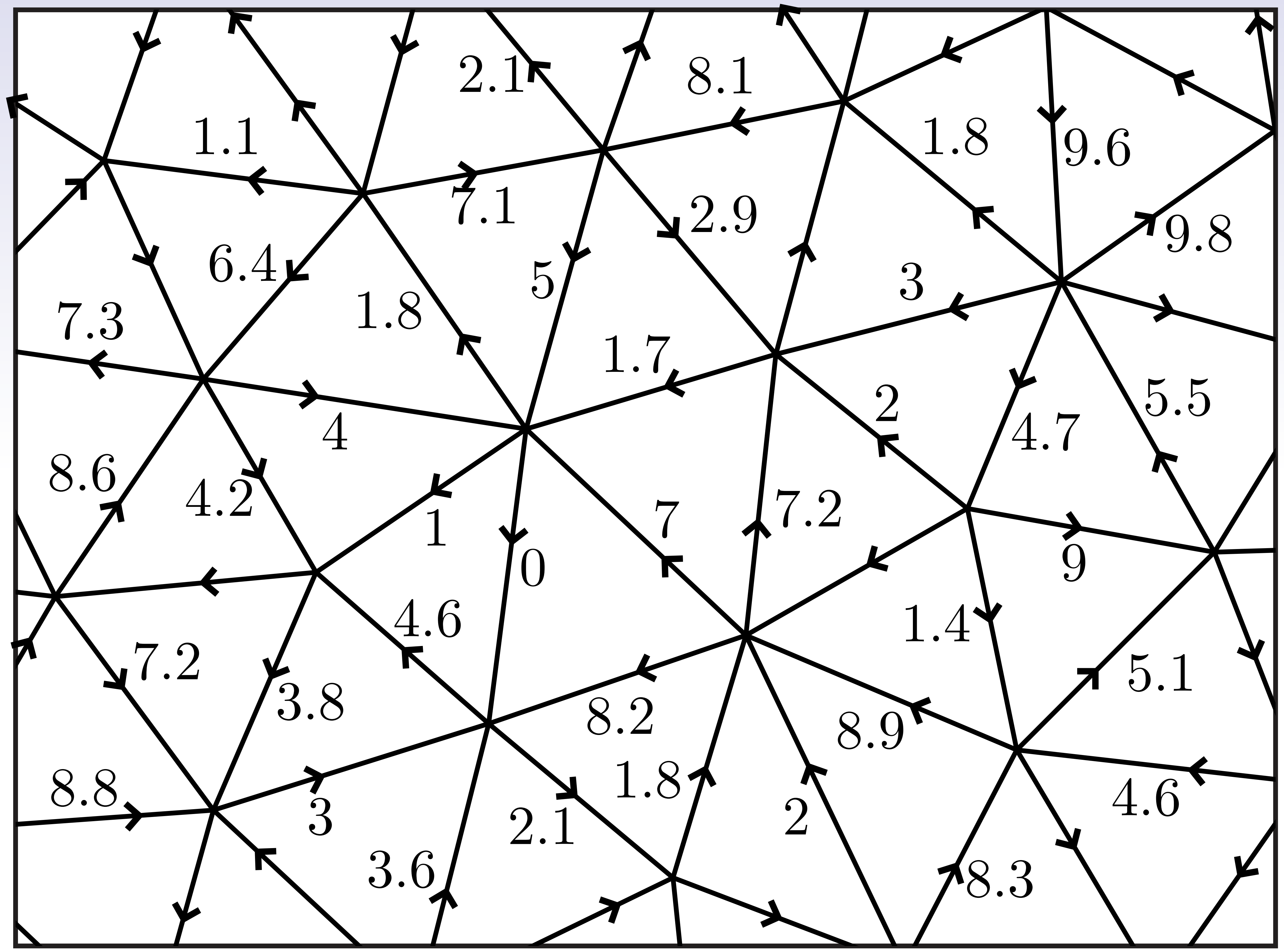
- Integration
  - break domain into small pieces
  - measure each piece with k-form
- Stokes' theorem
  - convert region integral to boundary integral
  - special cases: divergence theorem, F.T.C., *many more!*
  - will use *over and over* again in DEC/geometry processing

Discrete Exterior Calculus

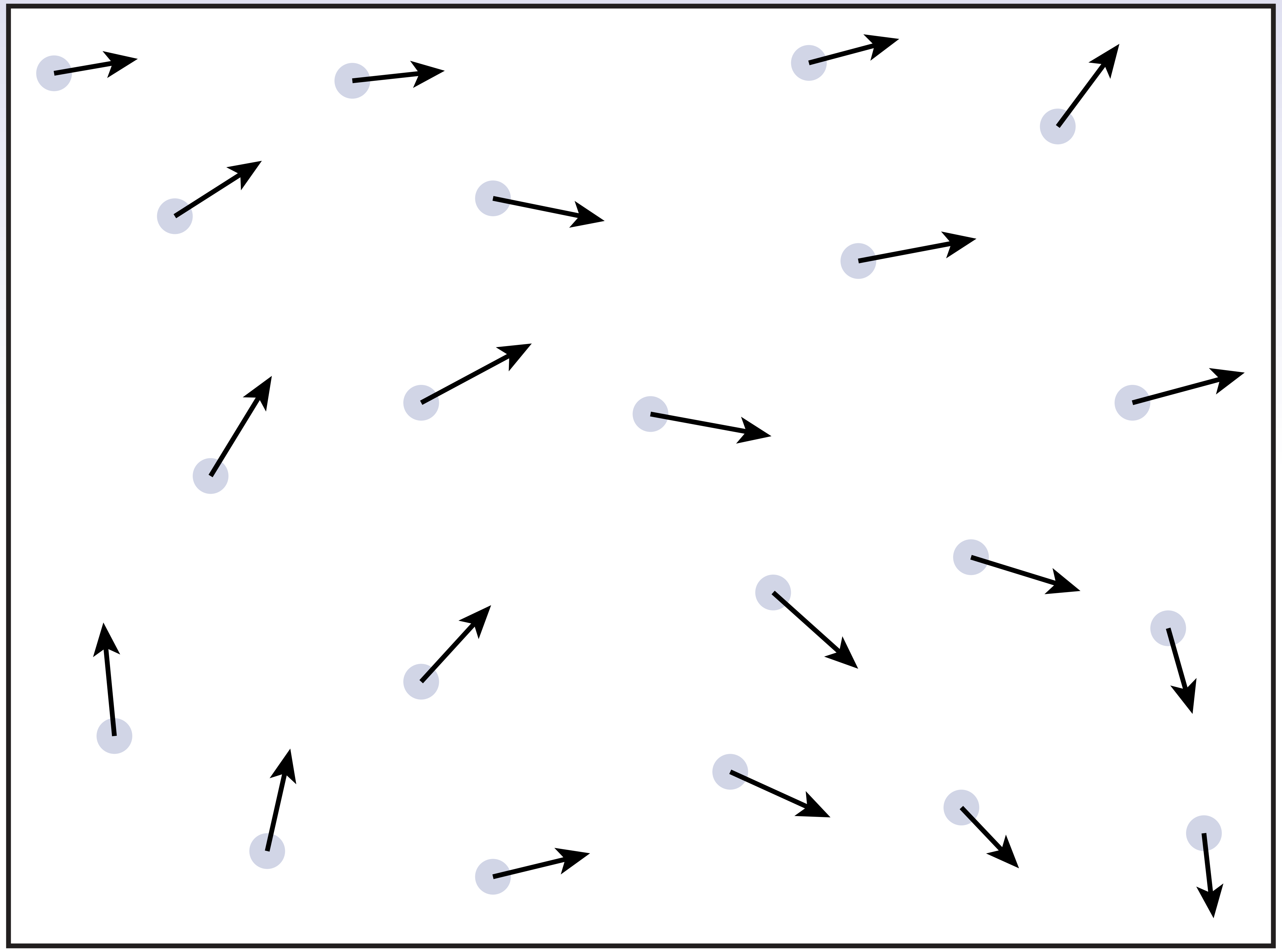


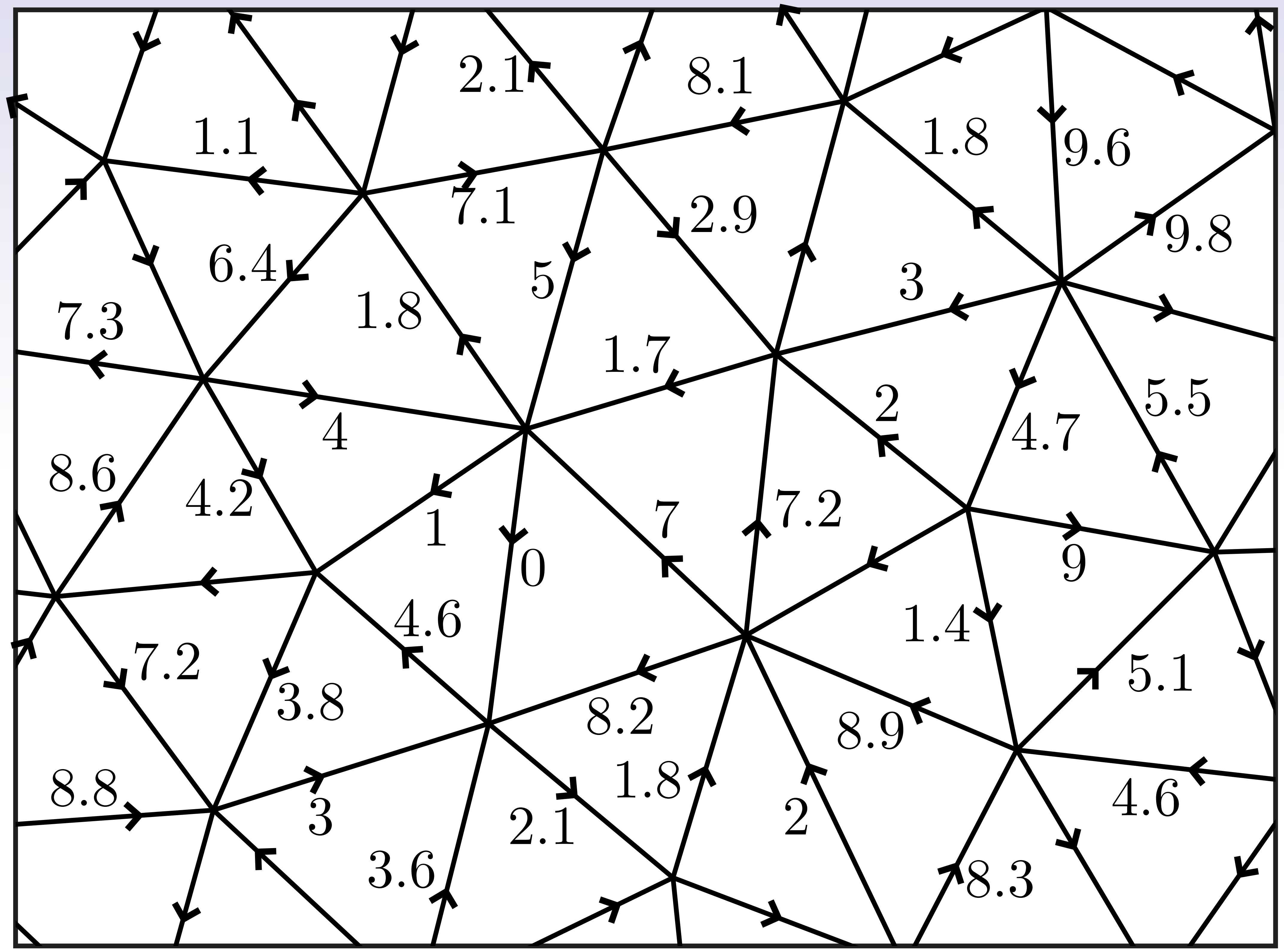
Discrete Differential Forms





Discrete Differential Forms

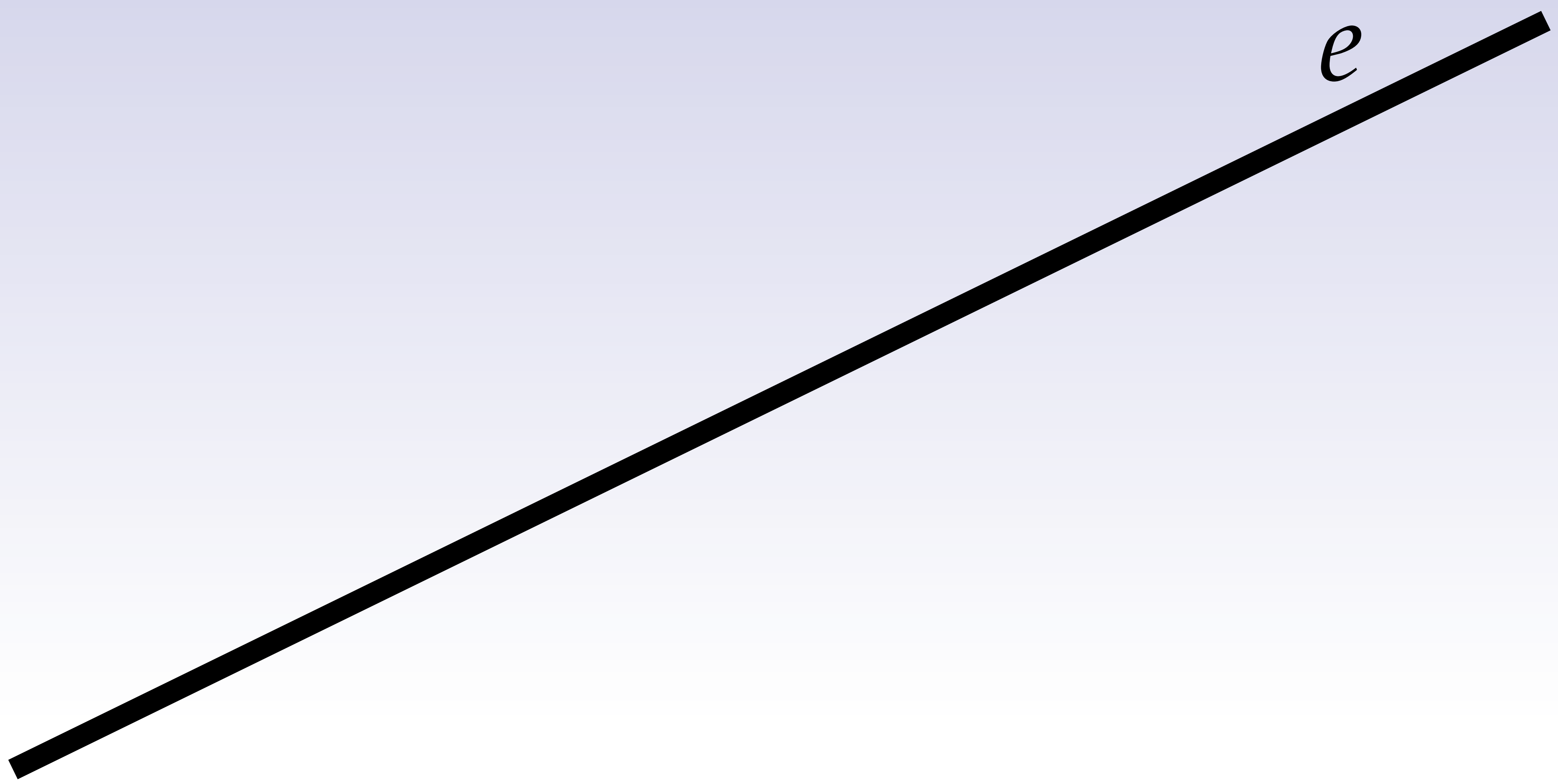




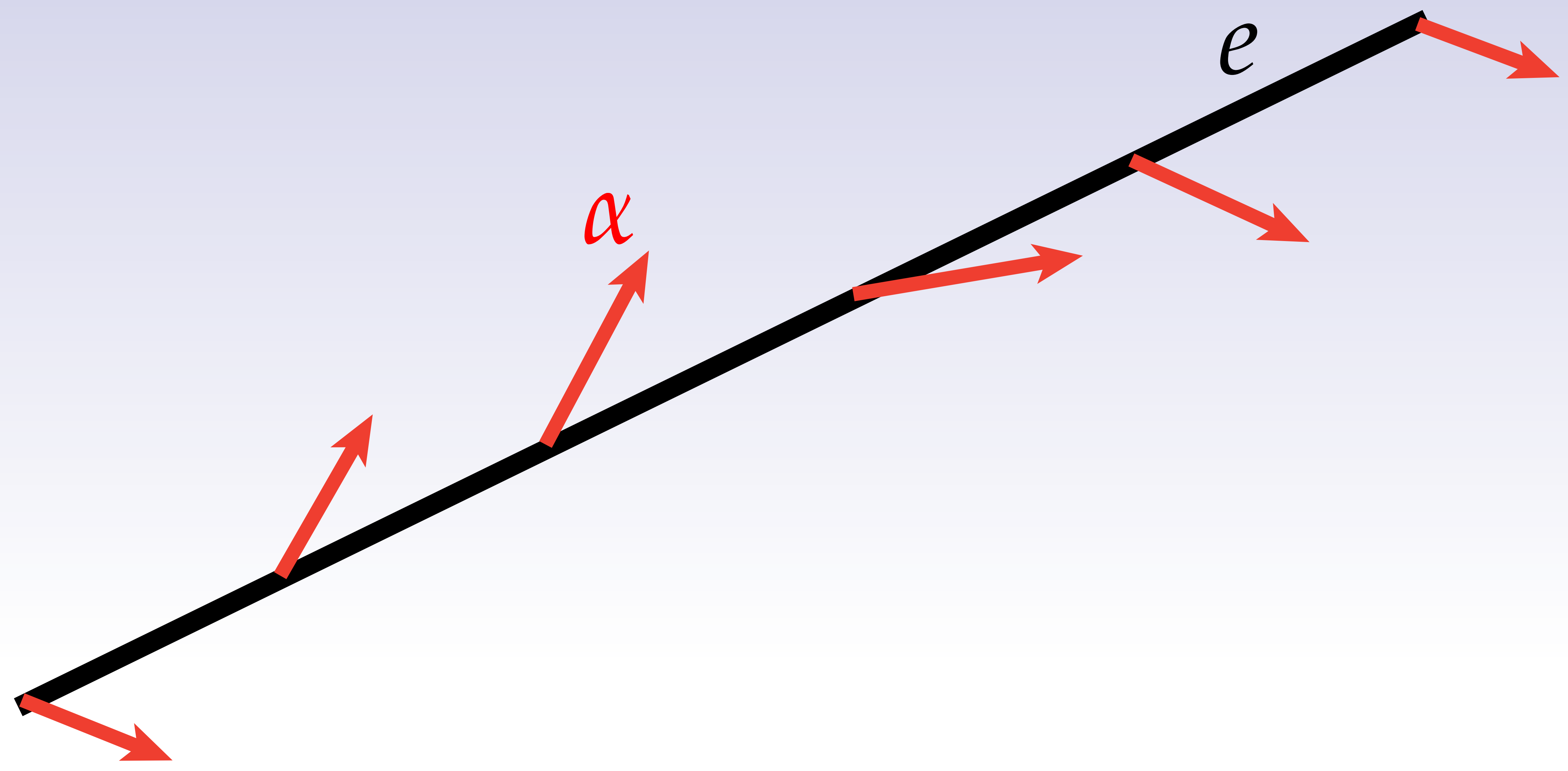
Basic idea: *integrate!*

Discrete 1-Form

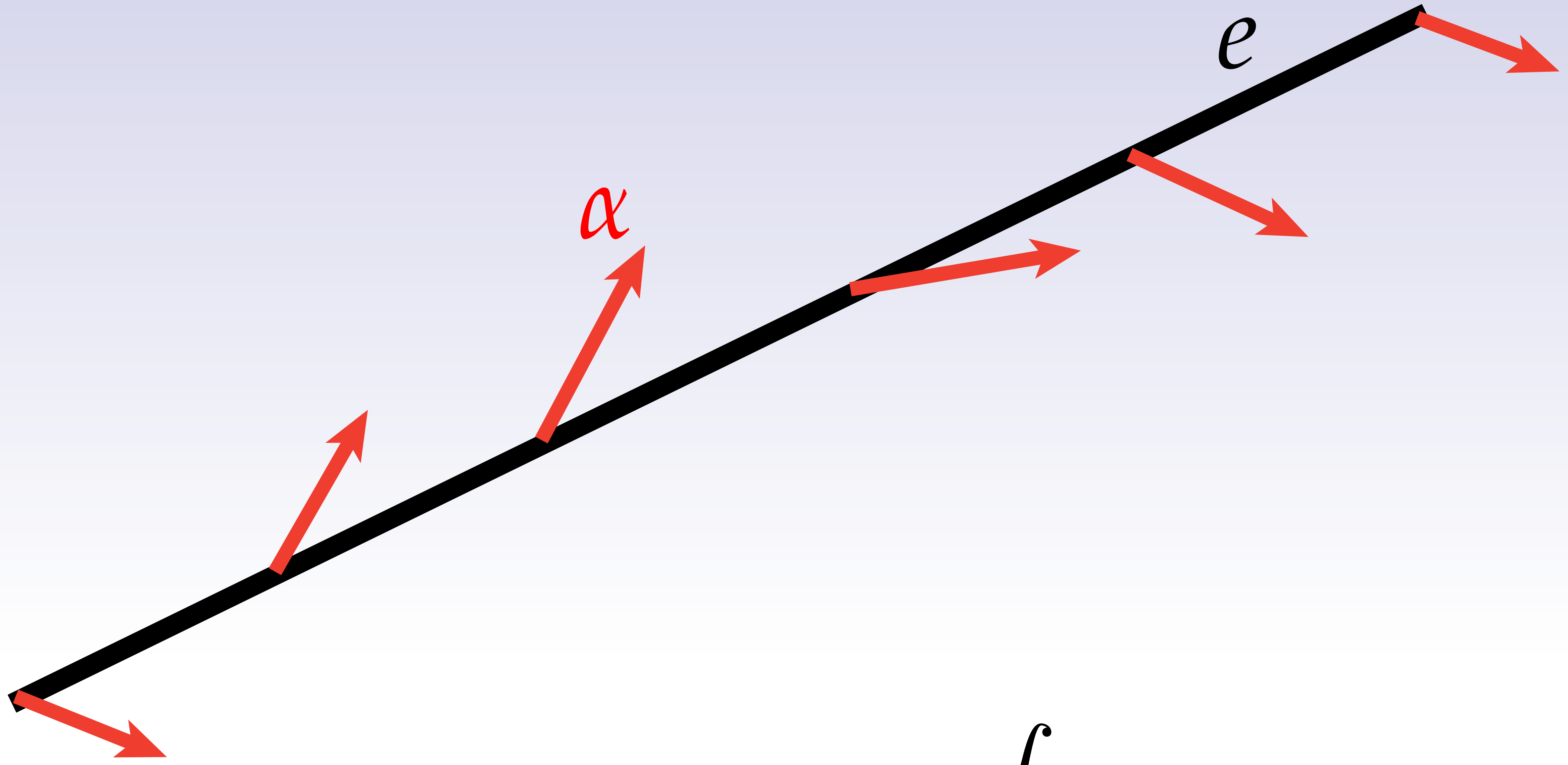
Discrete 1-Form



Discrete 1-Form

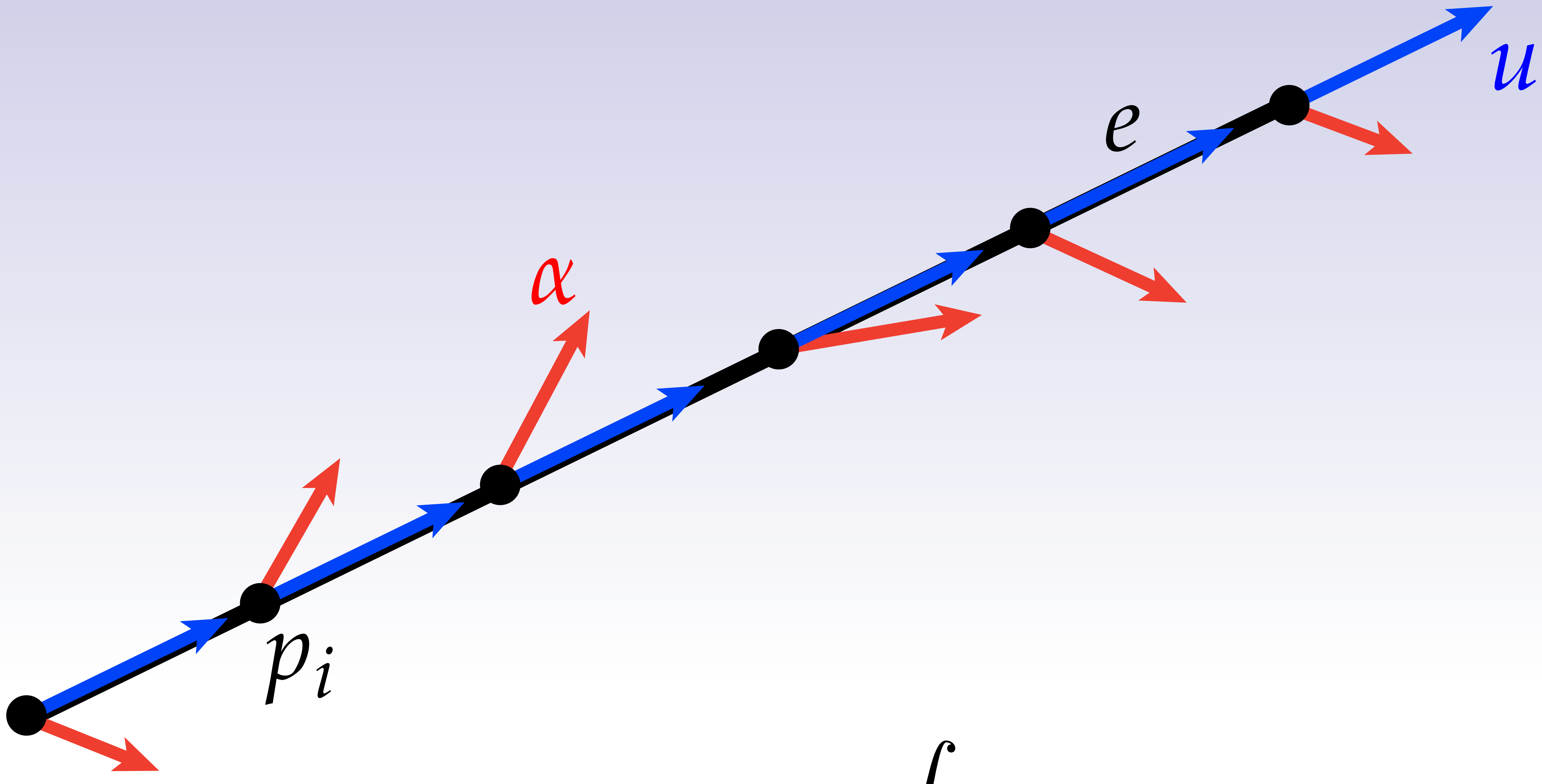


Discrete 1-Form



$$\hat{\alpha}_e := \int_e \alpha$$

Discrete 1-Form

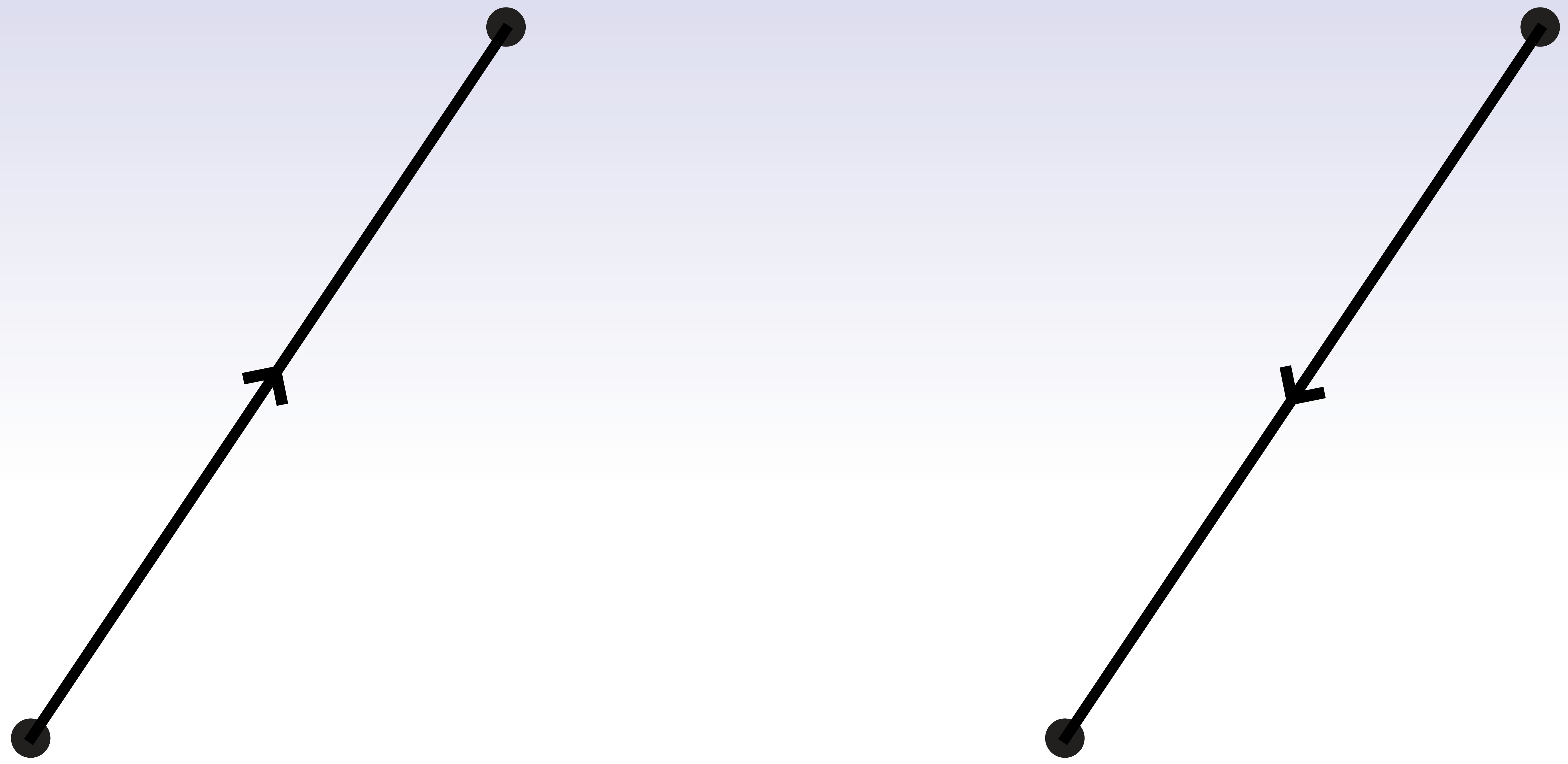


$$\hat{\alpha}_e := \int_e \alpha$$

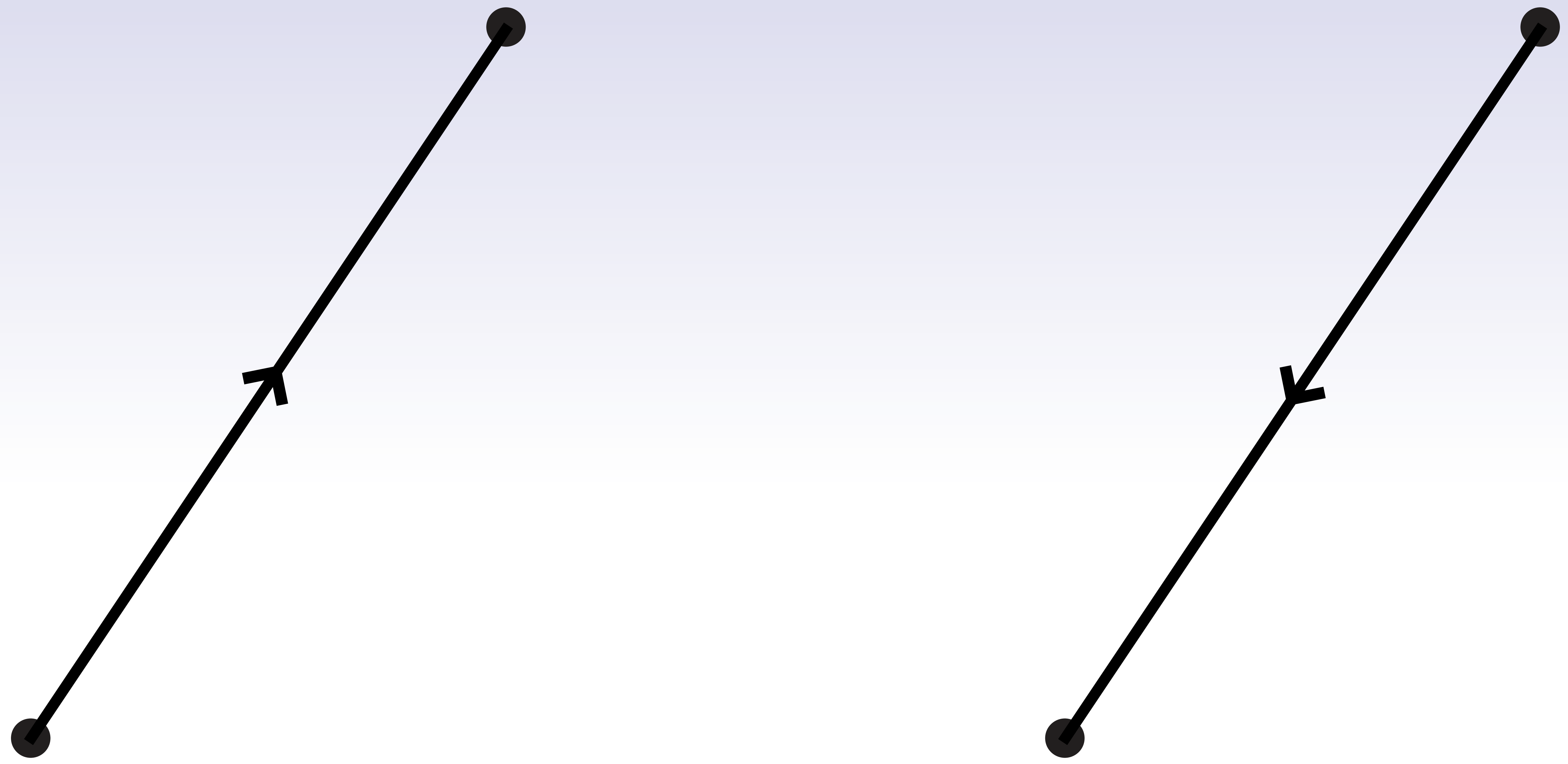
$$\int_e \alpha \approx |e| \left(\frac{1}{N} \sum_{i=1}^{N} \alpha_{p_i}(u) \right)$$

Orientation

Orientation

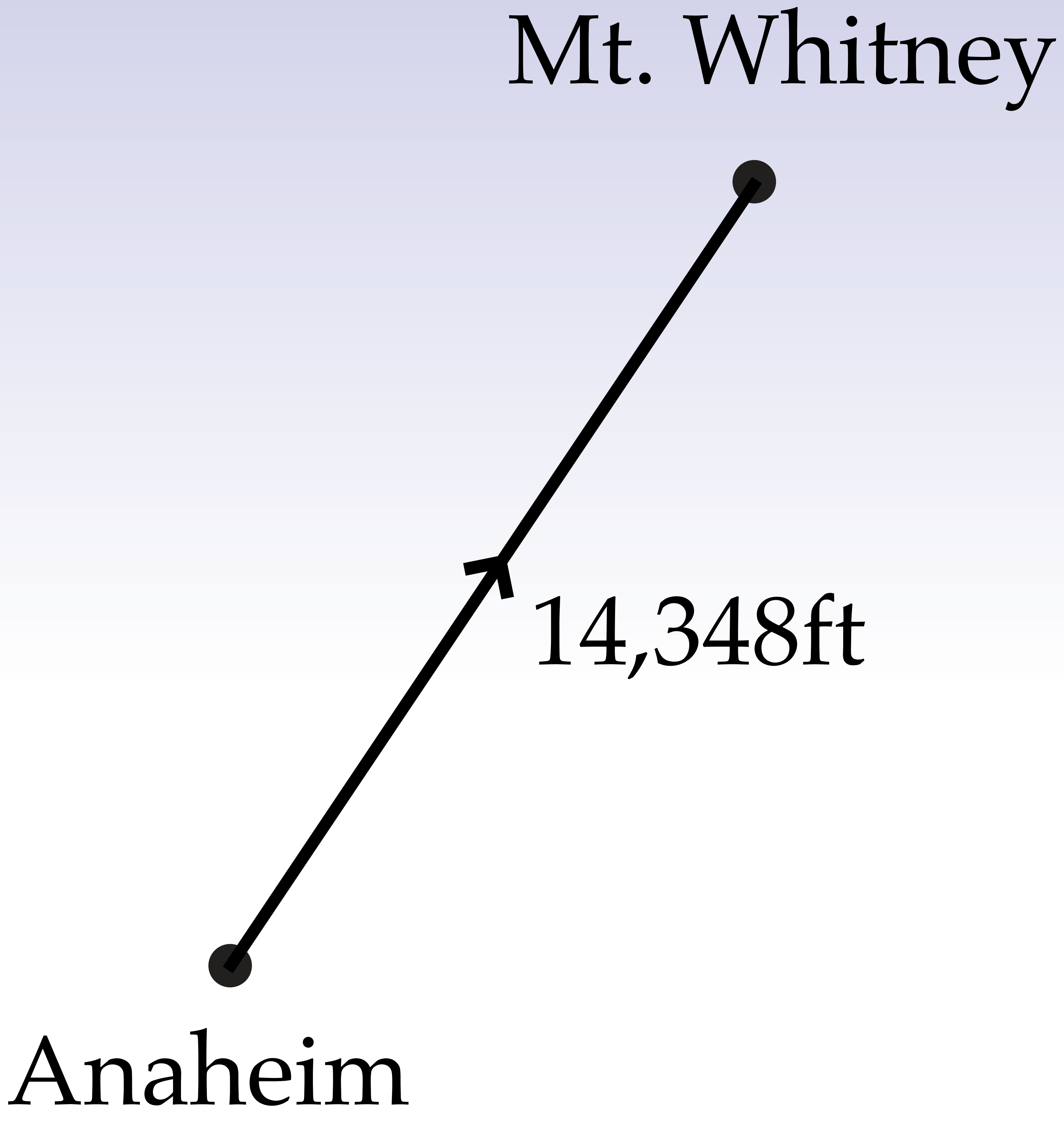


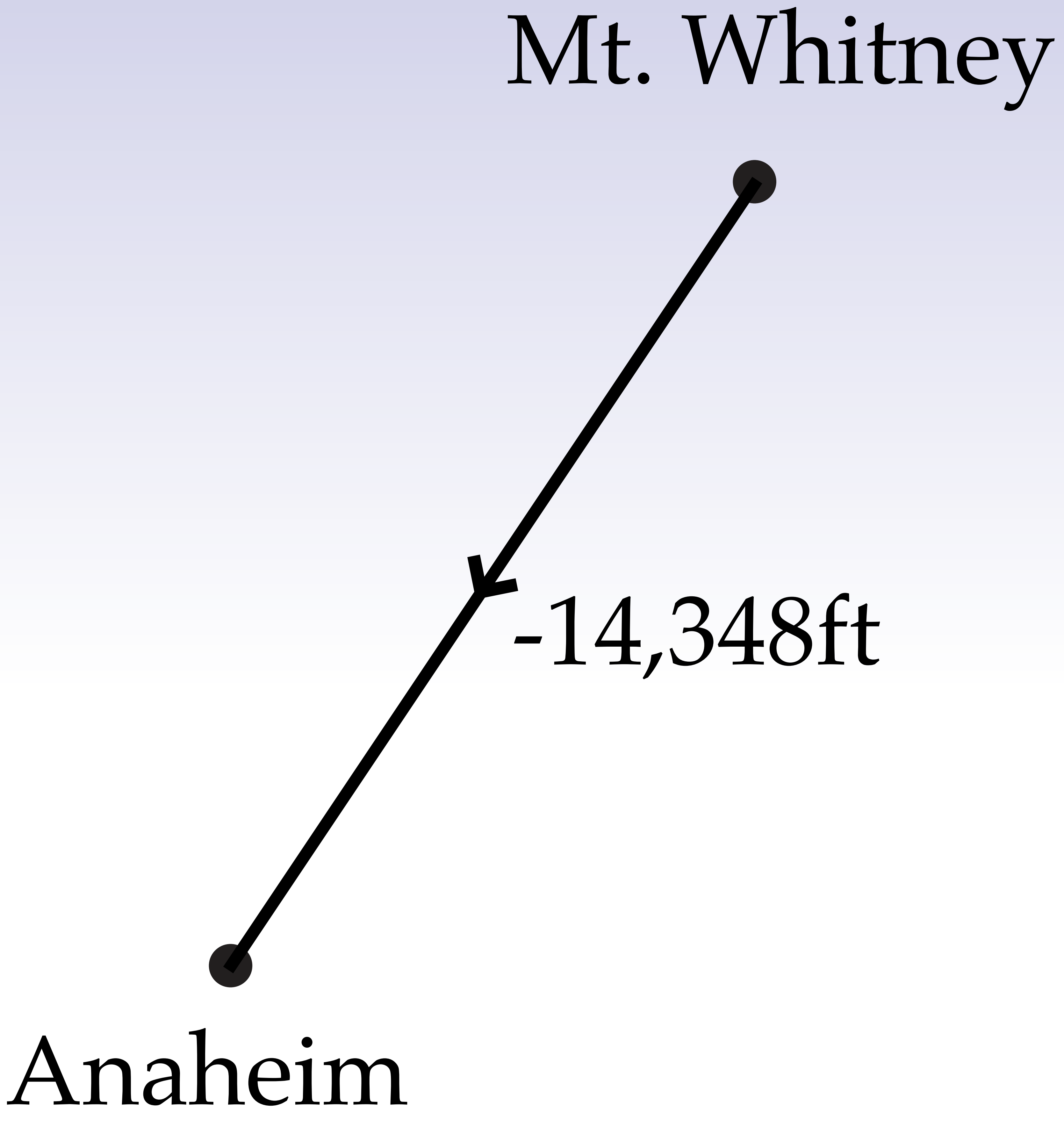
Orientation



$$\int_a^b \frac{\partial \phi}{\partial x} dx = \phi(b) - \phi(a) = -(\phi(a) - \phi(b)) = -\int_b^a \frac{\partial \phi}{\partial x} dx$$

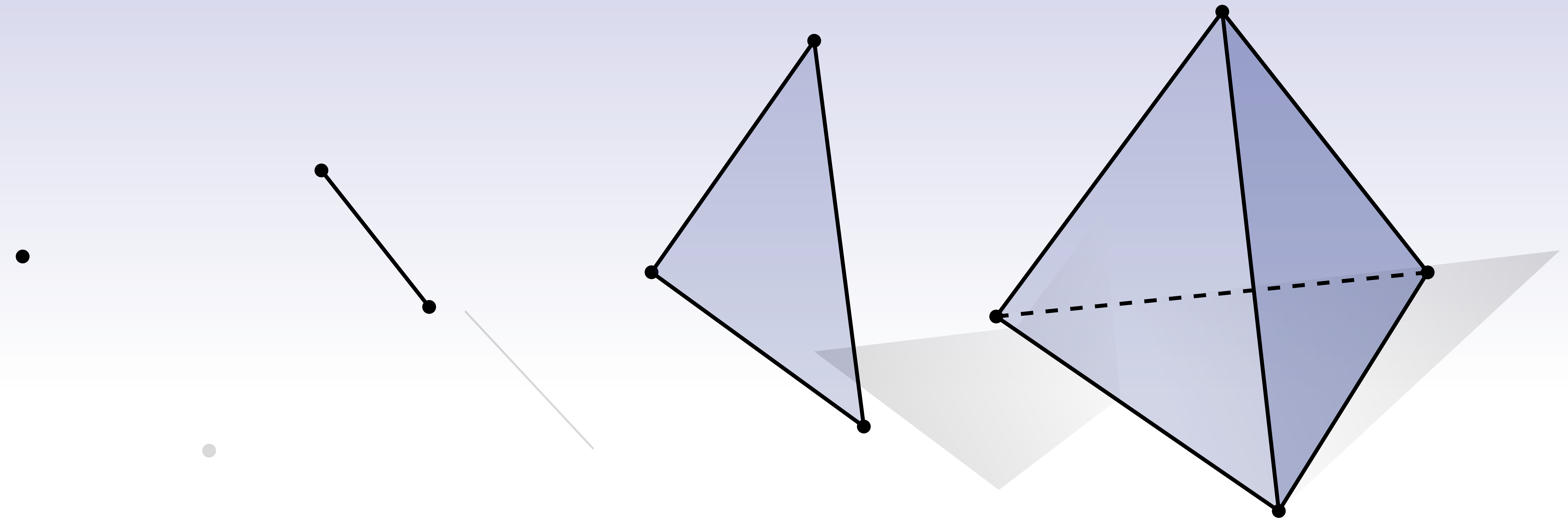
Orientation





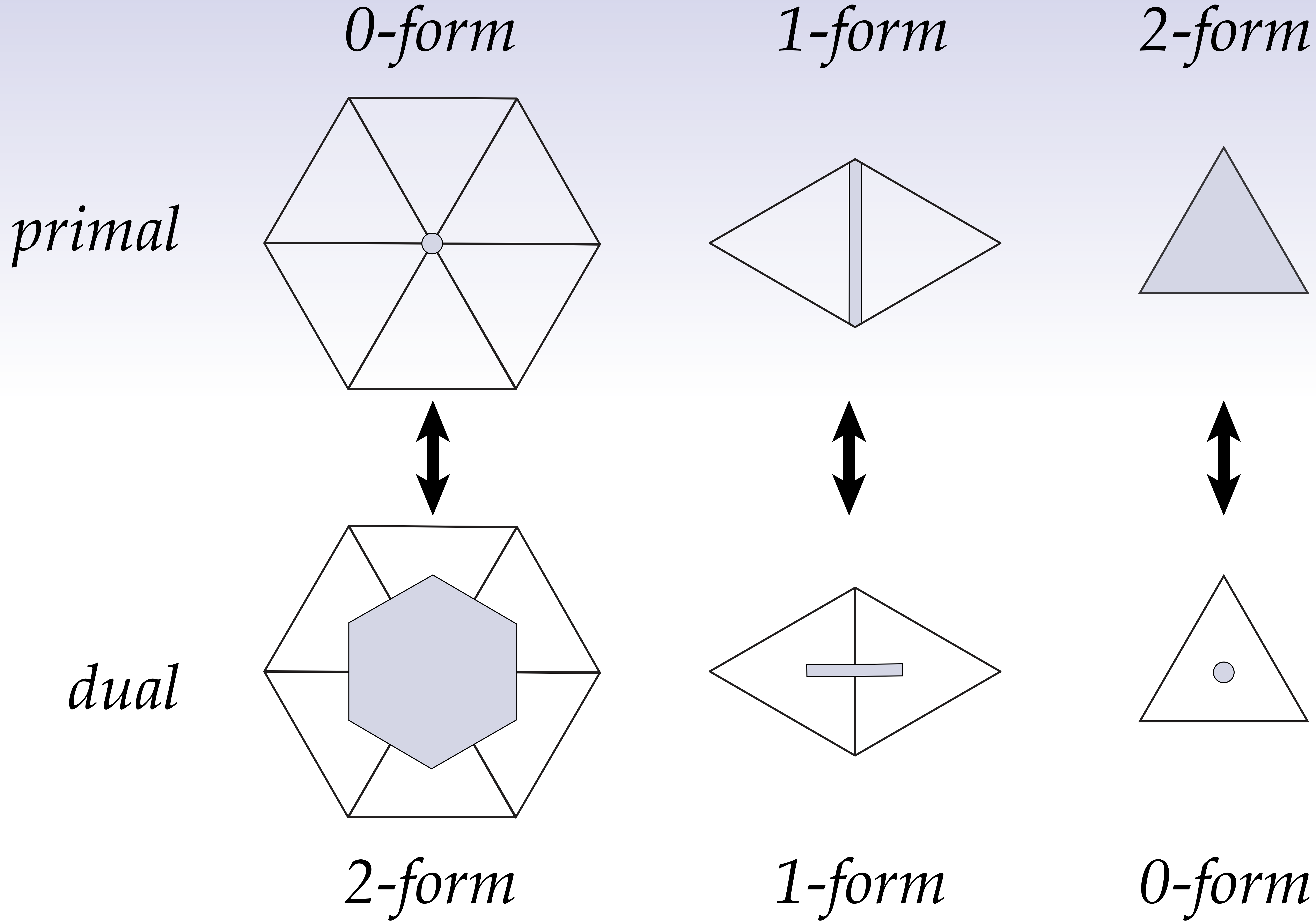
$$\int_a^b \frac{\partial \phi}{\partial x} dx = \phi(b) - \phi(a) = -(\phi(a) - \phi(b)) = -\int_b^a \frac{\partial \phi}{\partial x} dx$$

Discrete k-Forms



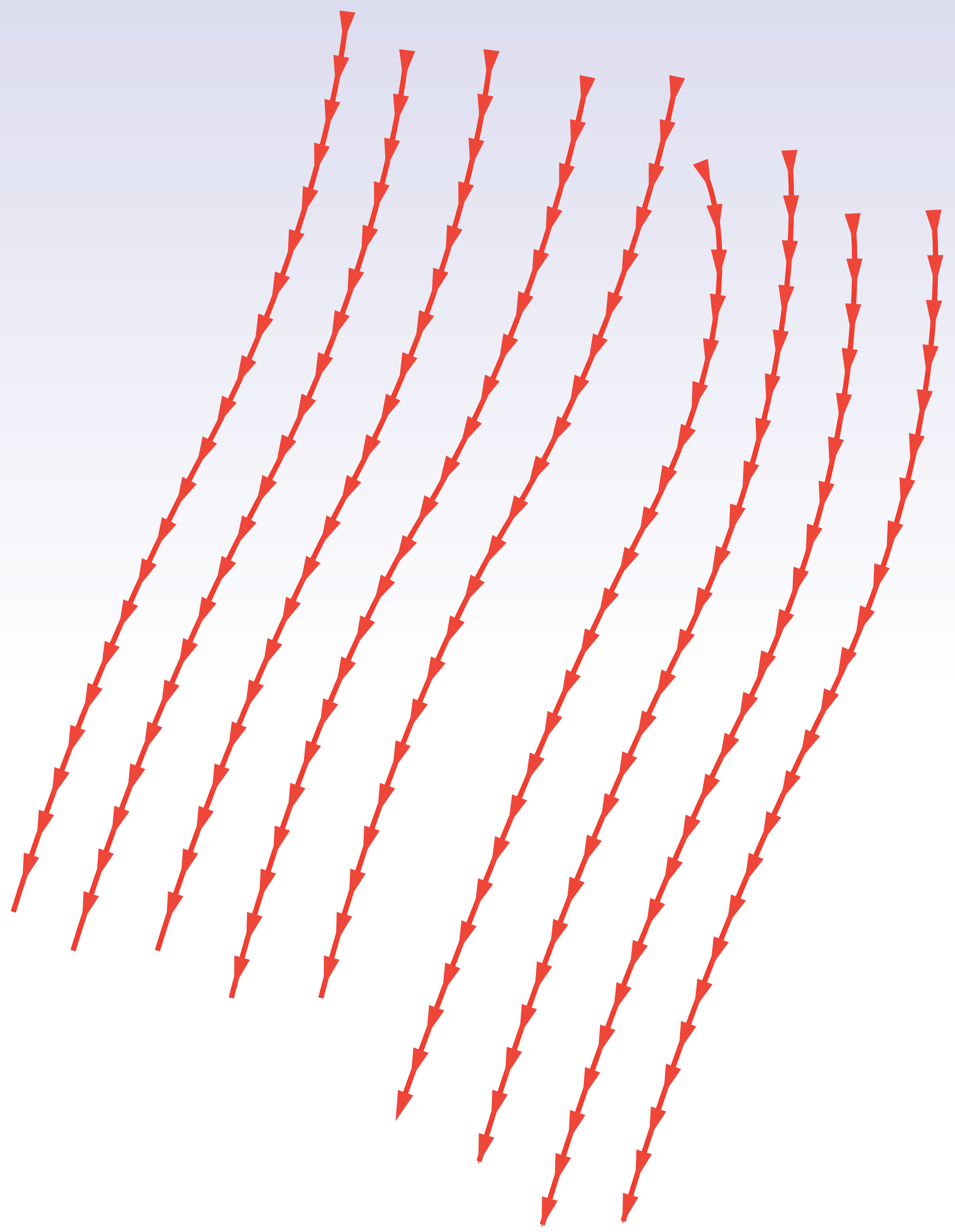
0-form	*1-form*	*2-form*	*3-form*
(vertices)	(edges)	(triangles)	(tetrahedra)
$\hat{\phi} \in \mathbb{R}^{\|V\|}$	$\hat{\alpha} \in \mathbb{R}^{\|E\|}$	$\hat{\beta} \in \mathbb{R}^{\|F\|}$	$\hat{\omega} \in \mathbb{R}^{\|T\|}$

Primal vs. Dual

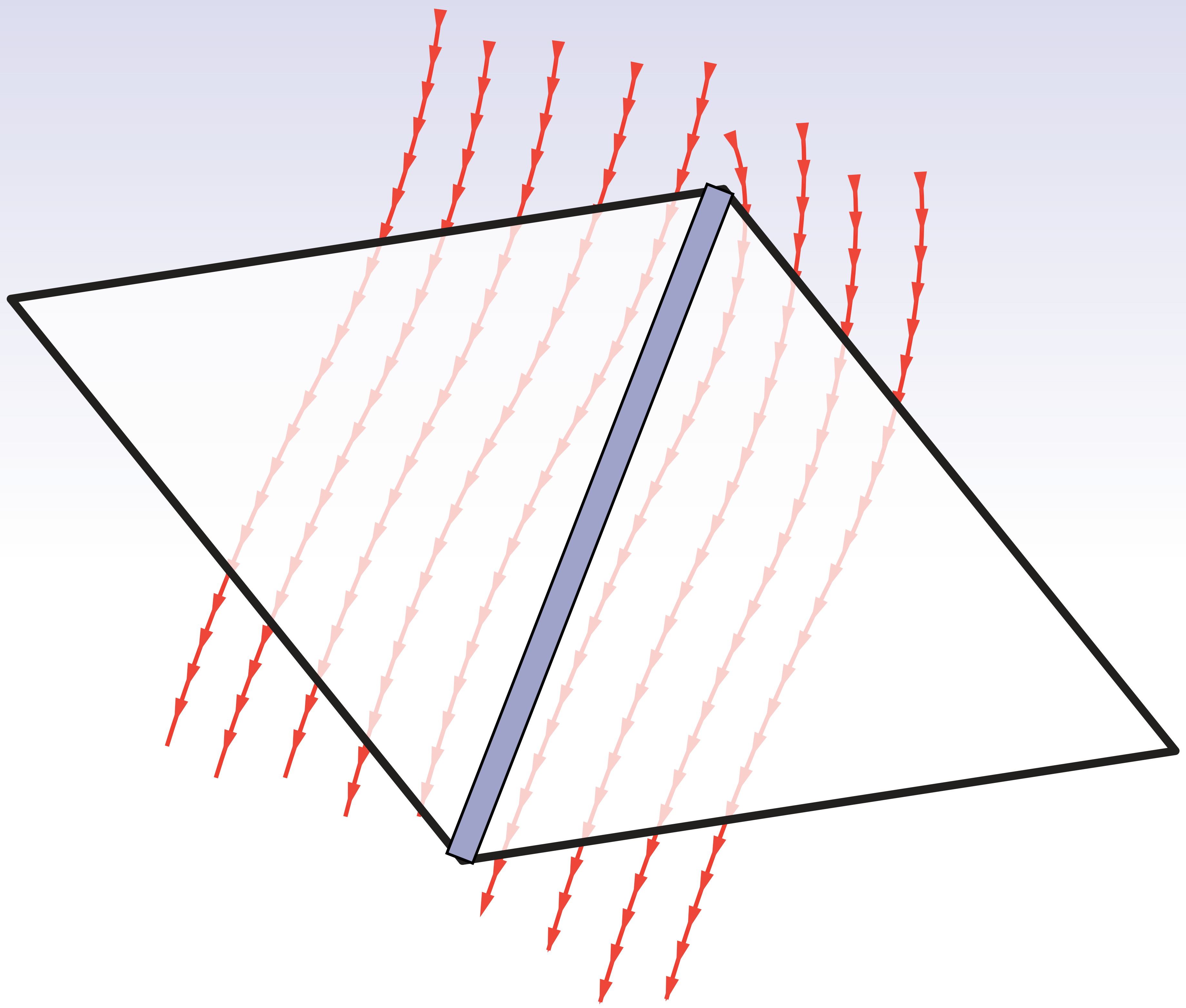


Discrete Hodge Star

Discrete Hodge Star

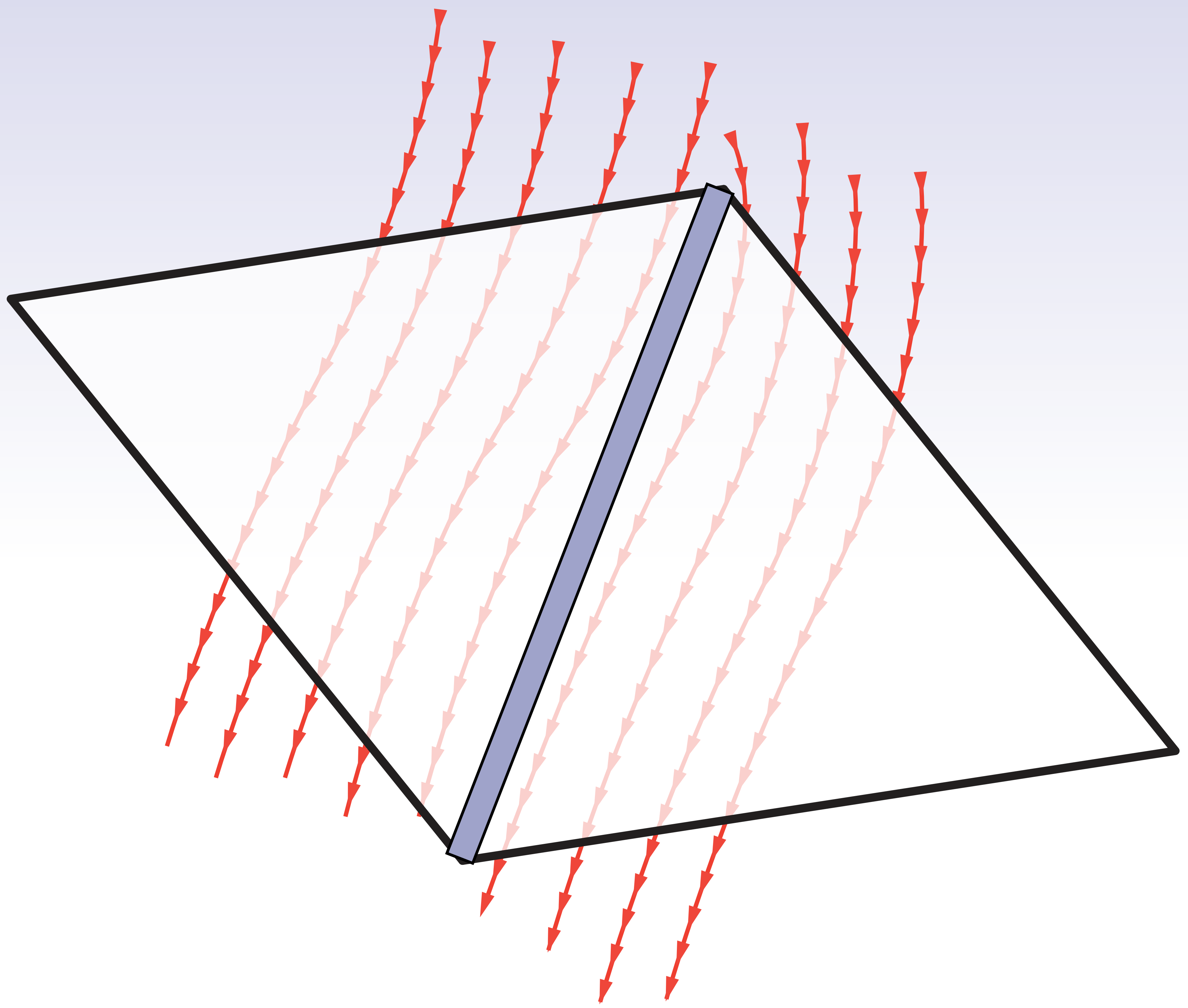


Discrete Hodge Star

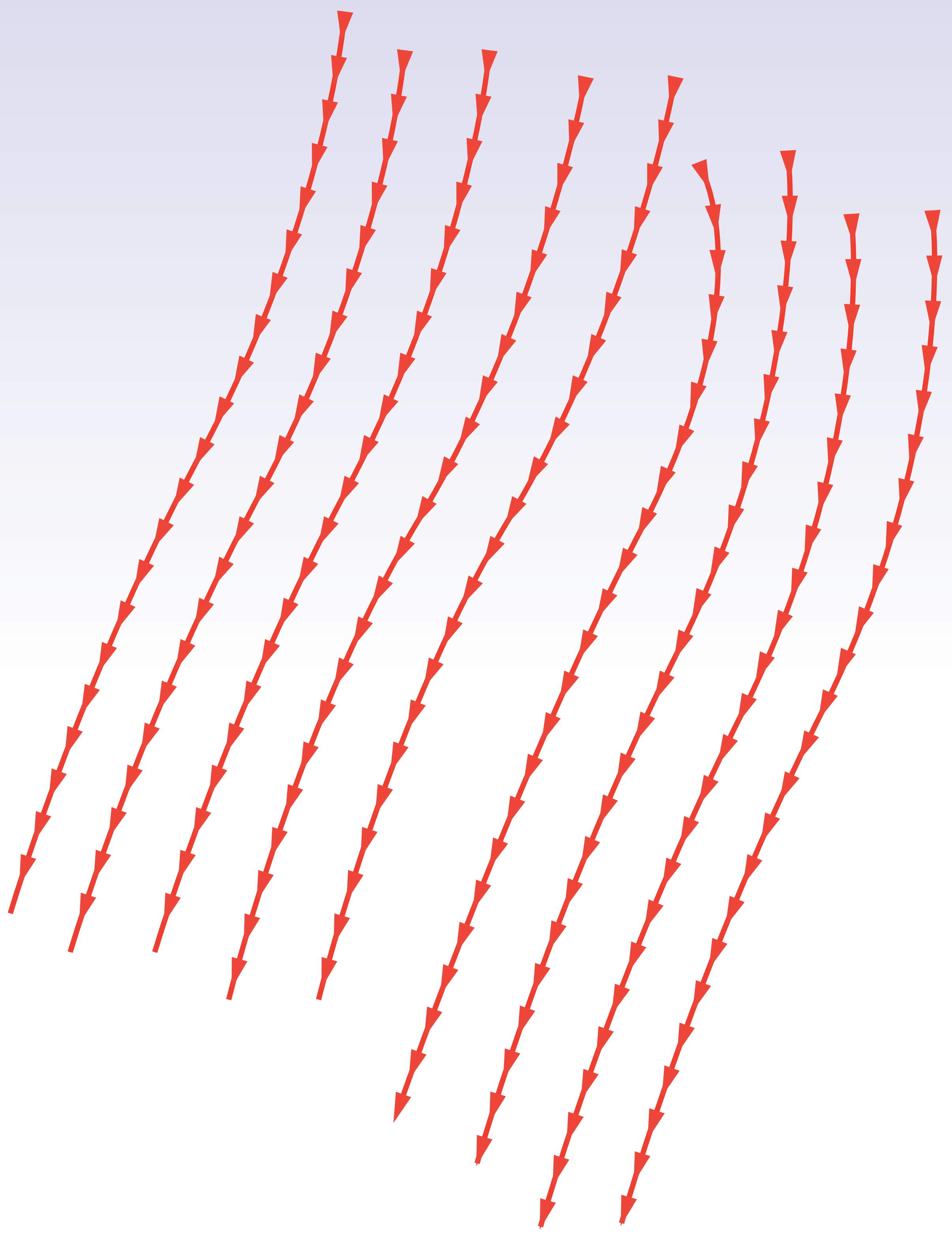


primal 1-form
(circulation)

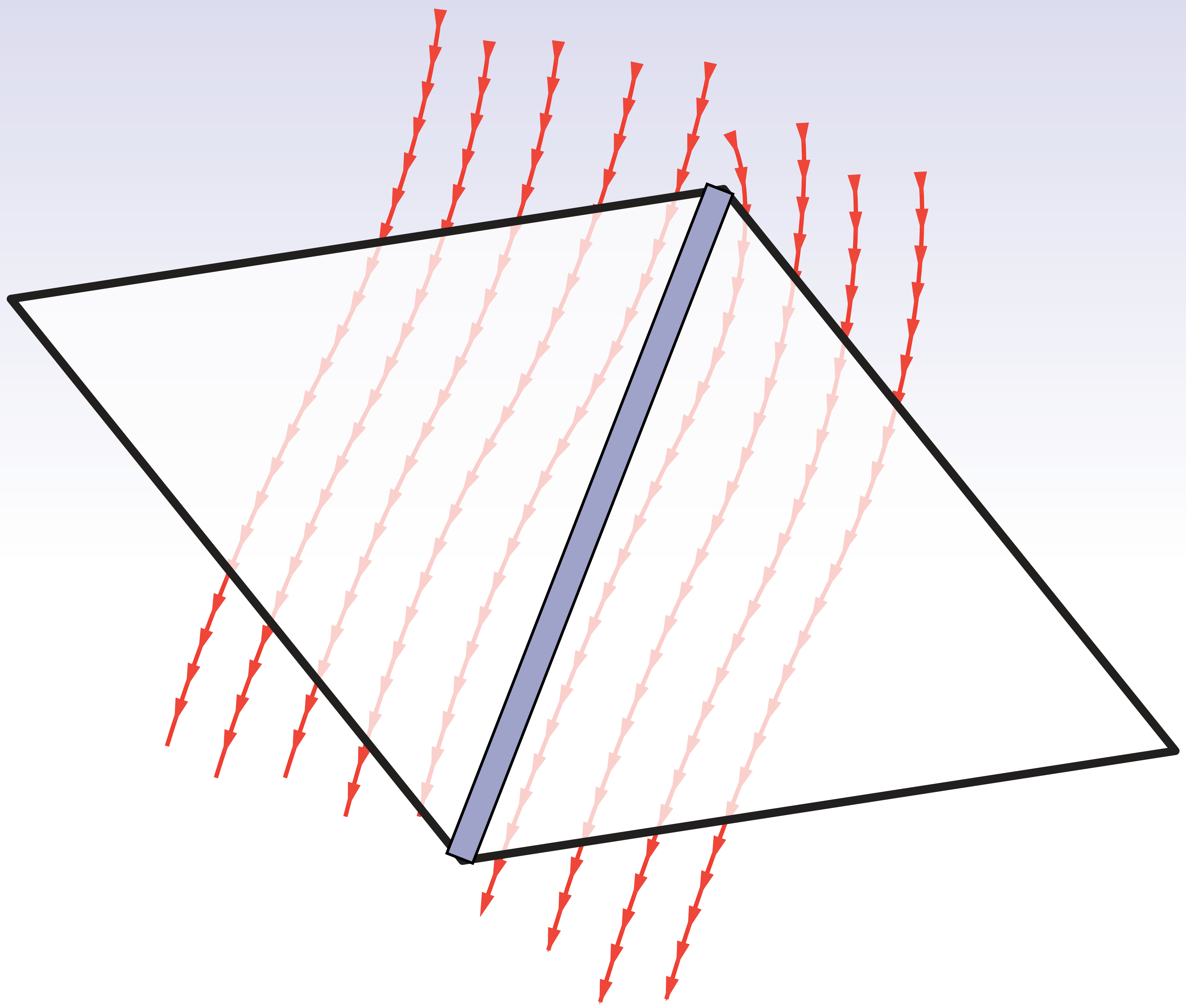
Discrete Hodge Star



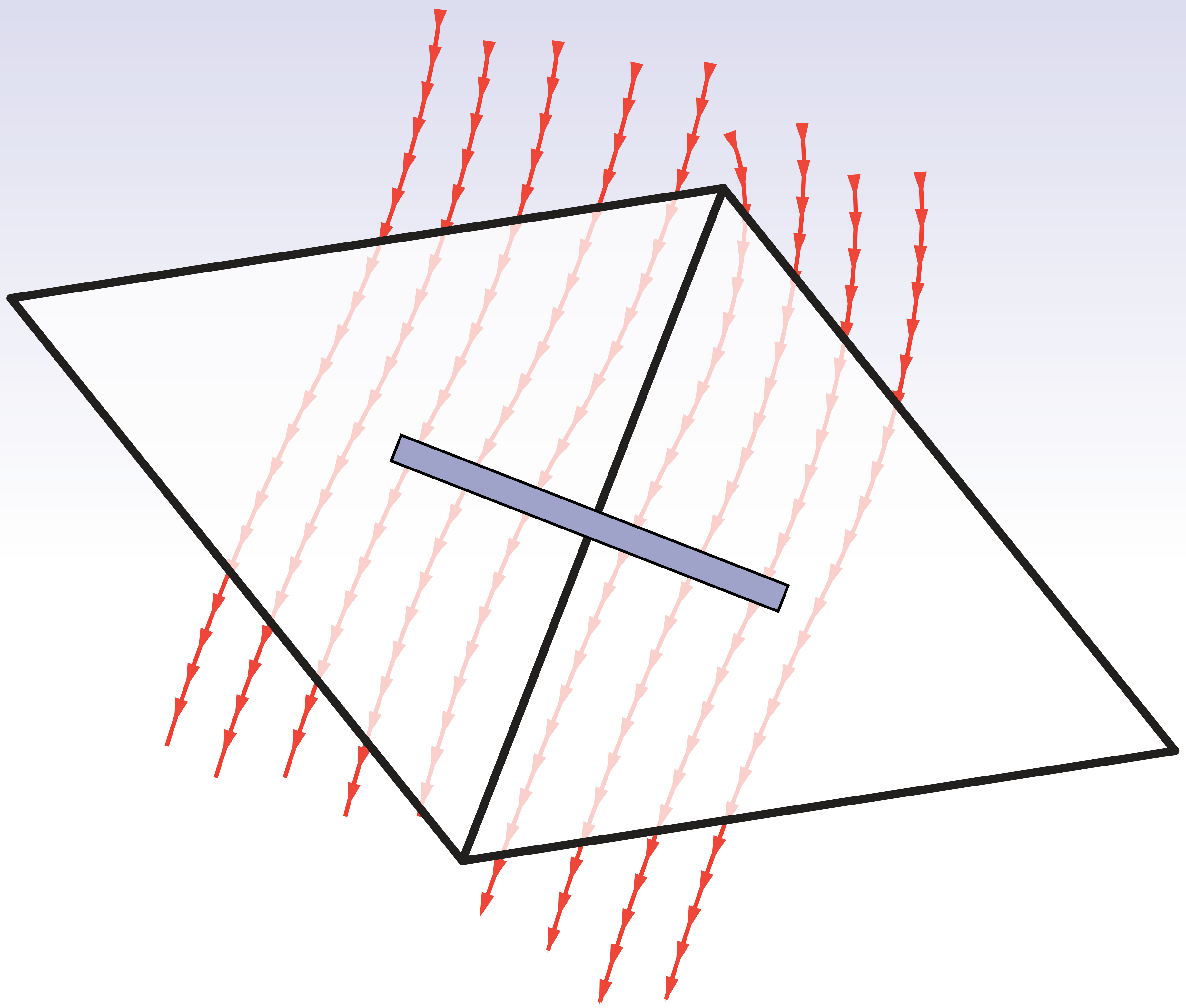
primal 1-form
(circulation)



Discrete Hodge Star

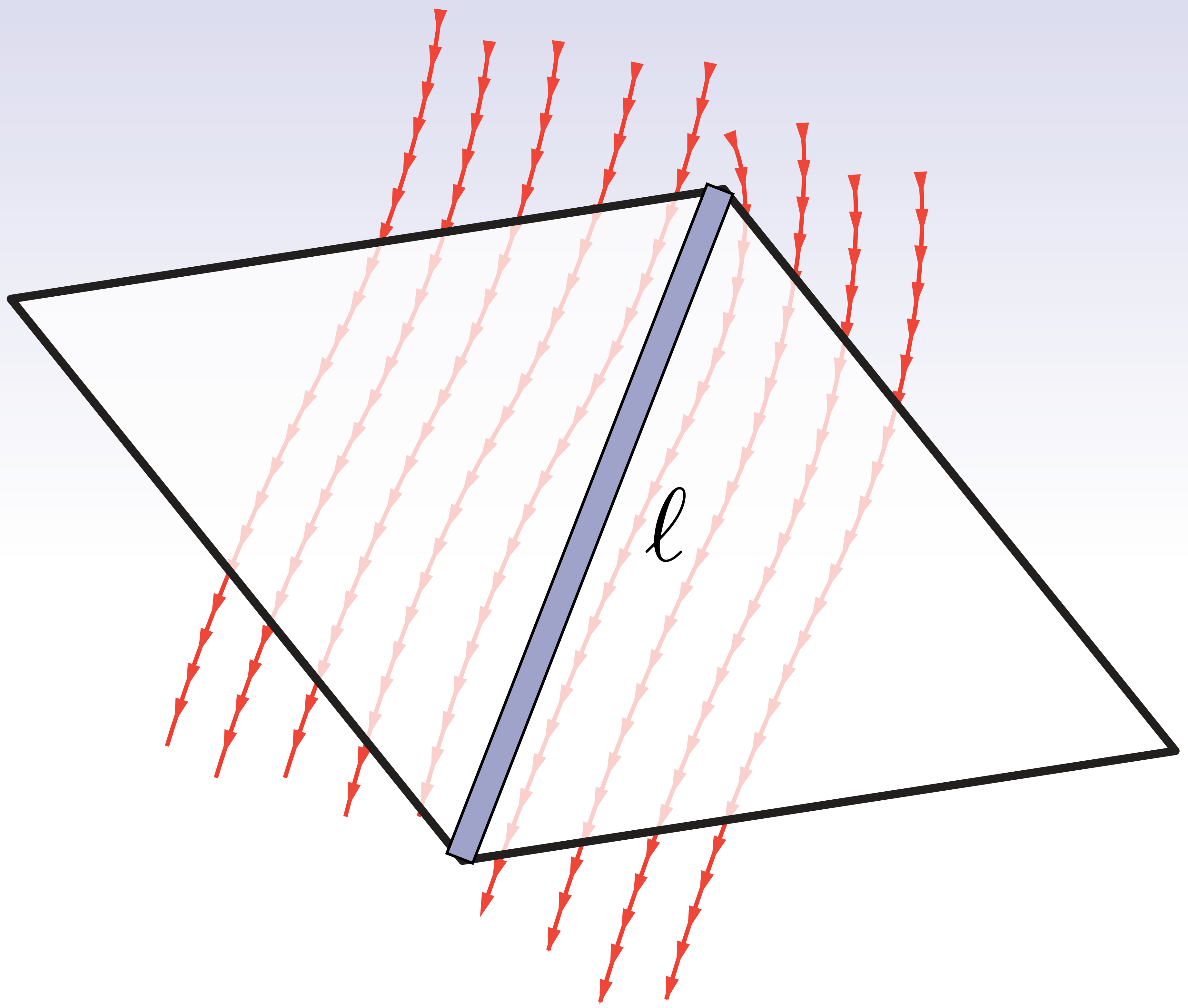


primal 1-form
(circulation)

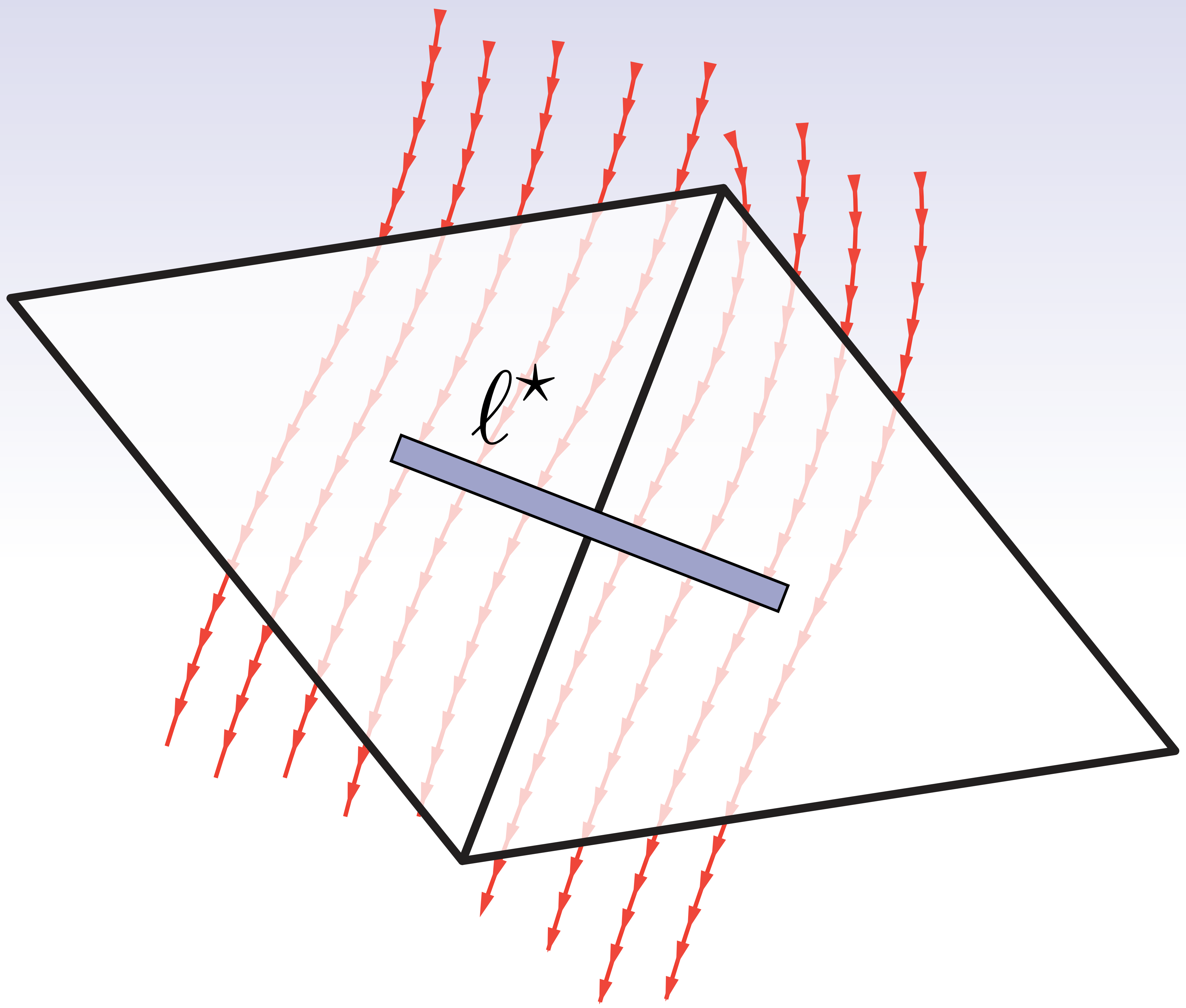


dual 1-form
(flux)

Discrete Hodge Star

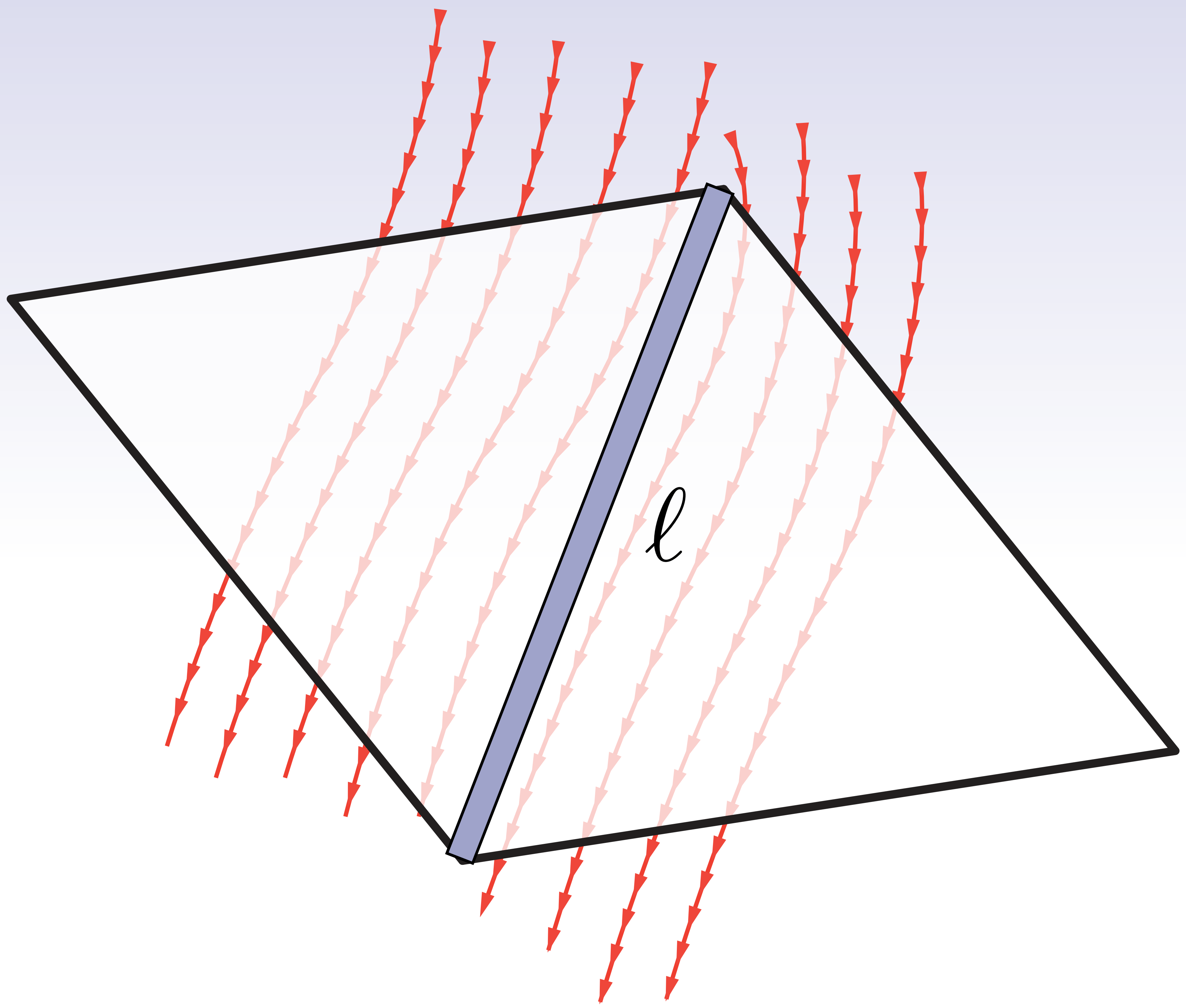


primal 1-form
(circulation)



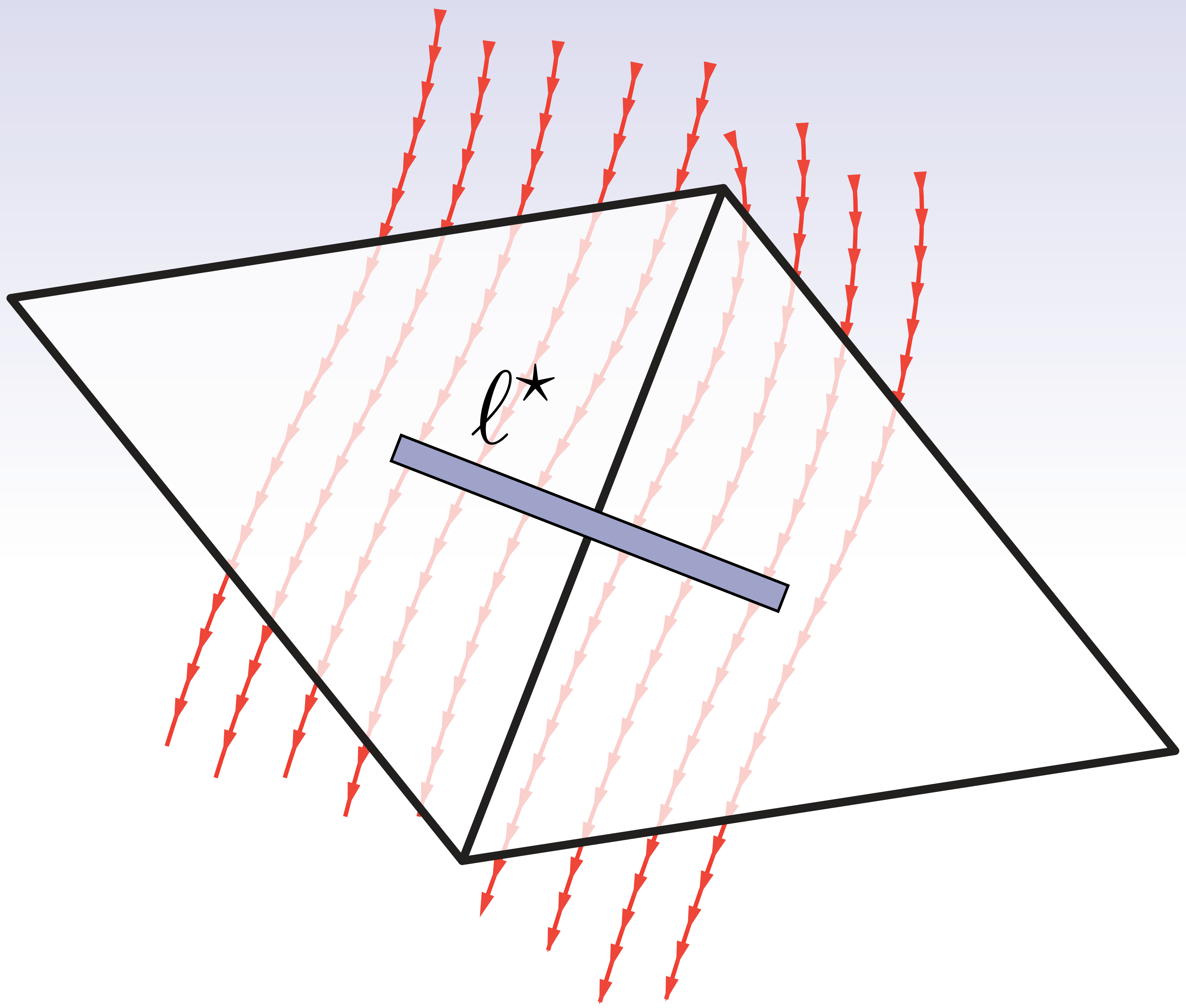
dual 1-form
(flux)

Discrete Hodge Star



primal 1-form
(circulation)

$$\star\hat{\alpha}_e = \frac{\ell^\star}{\ell}\hat{\alpha}$$

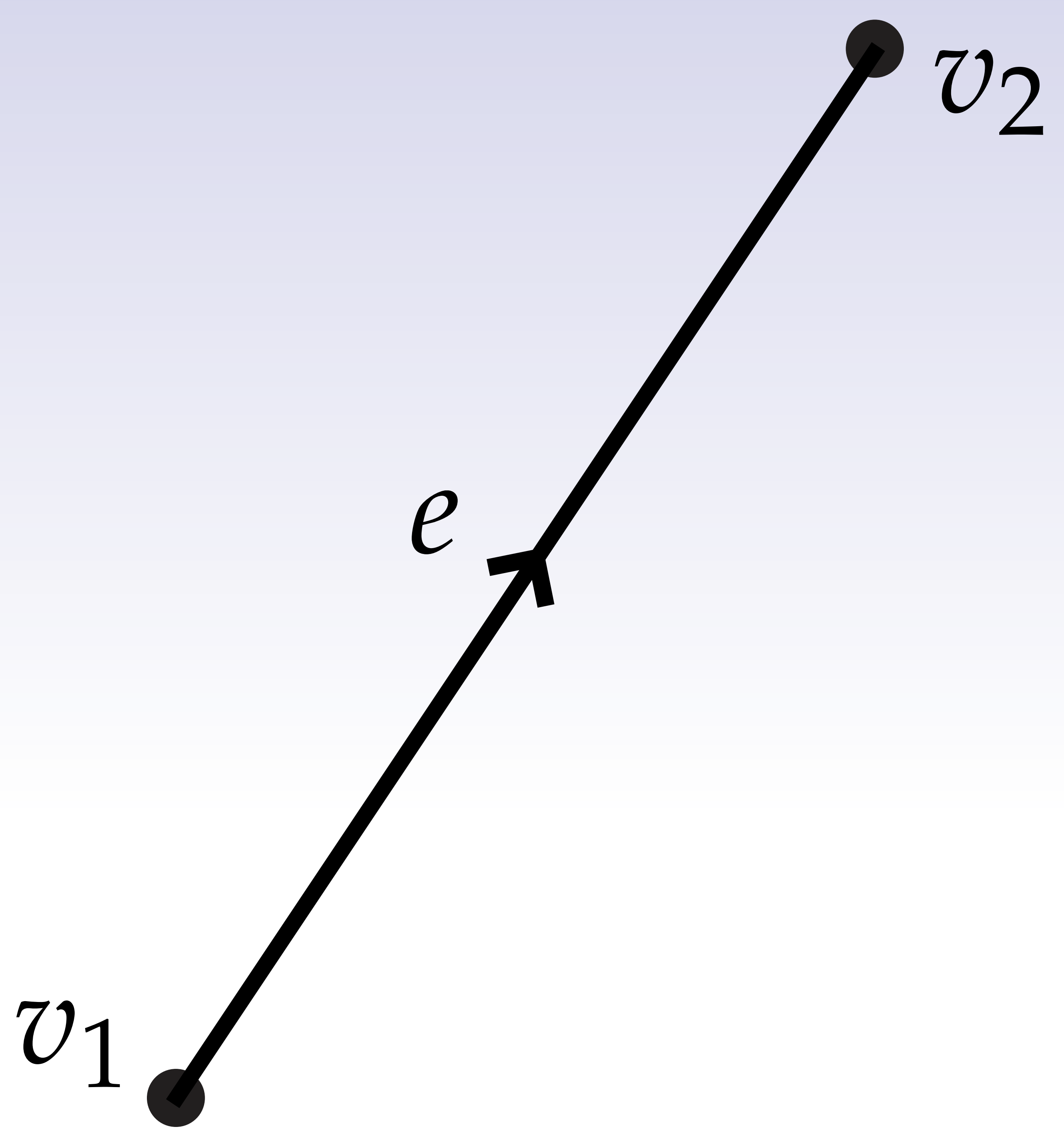


dual 1-form
(flux)

Discrete Exterior Derivative (0-Forms)

ϕ - *primal 0-form* (vertices)

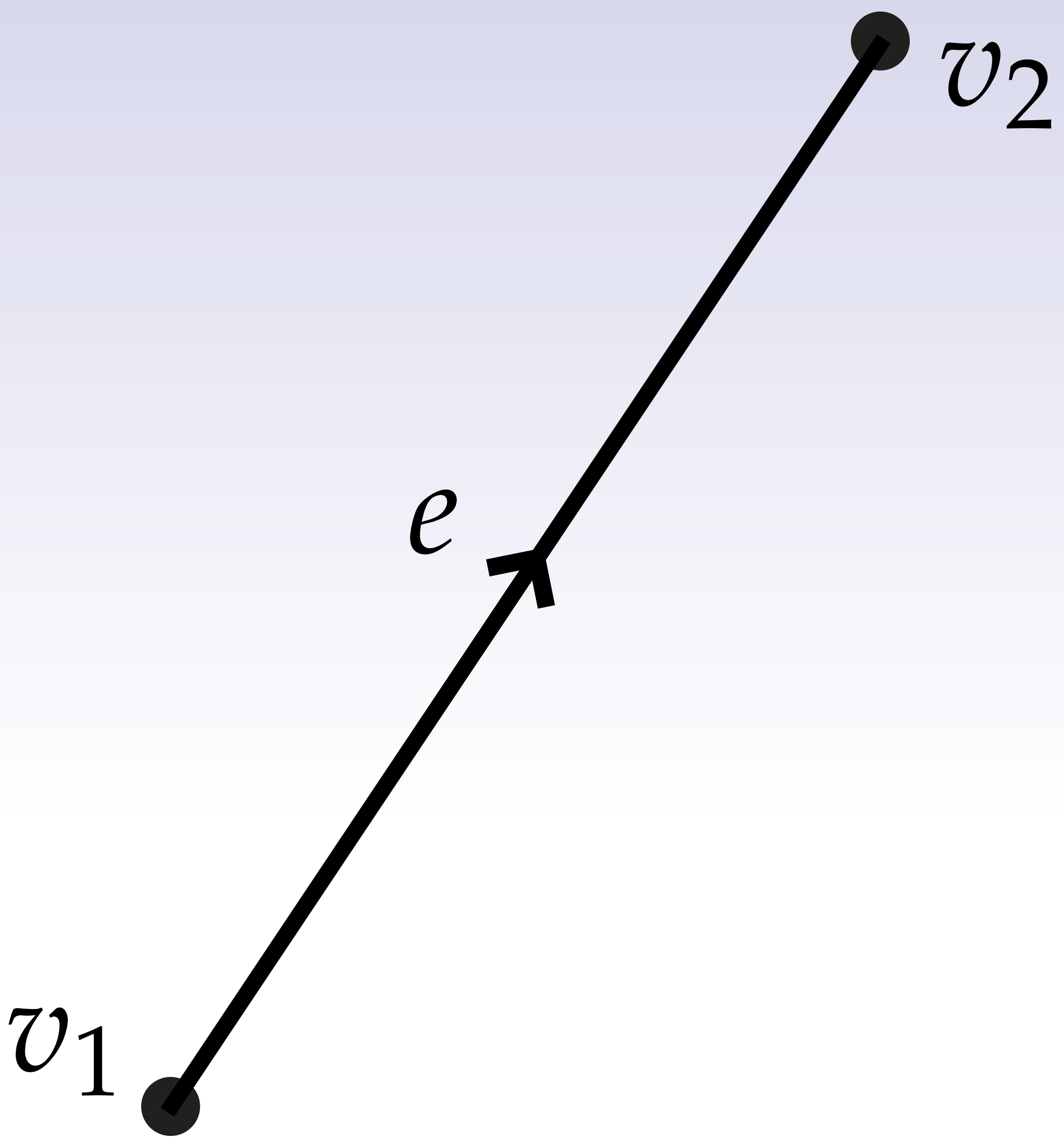
$d\phi$ - *primal 1-form* (edges)



Discrete Exterior Derivative (0-Forms)

ϕ - *primal 0-form* (vertices)

$d\phi$ - *primal 1-form* (edges)

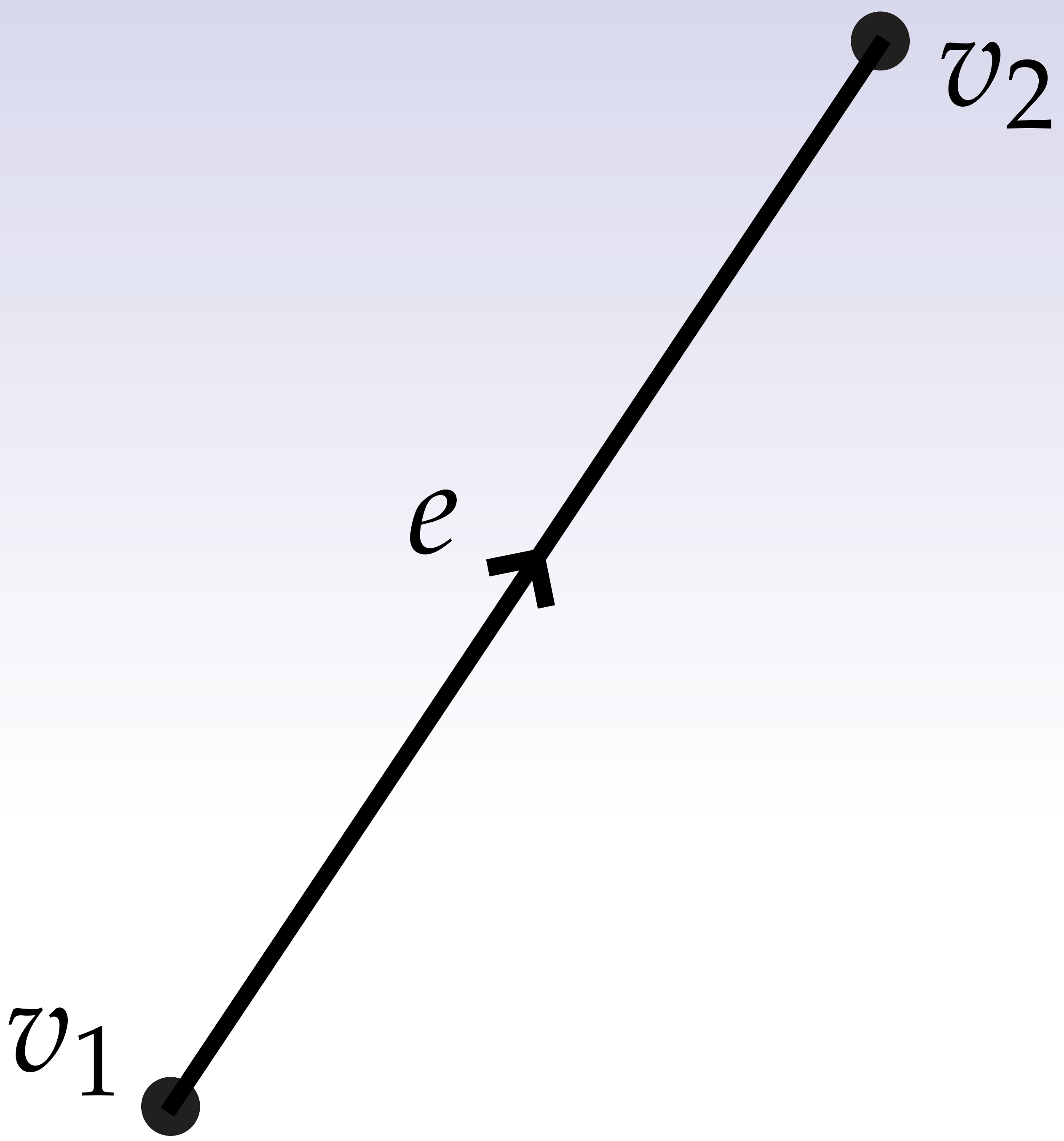


$$(\widehat{d\phi})_e$$

Discrete Exterior Derivative (0-Forms)

ϕ - *primal 0-form* (vertices)

$d\phi$ - *primal 1-form* (edges)

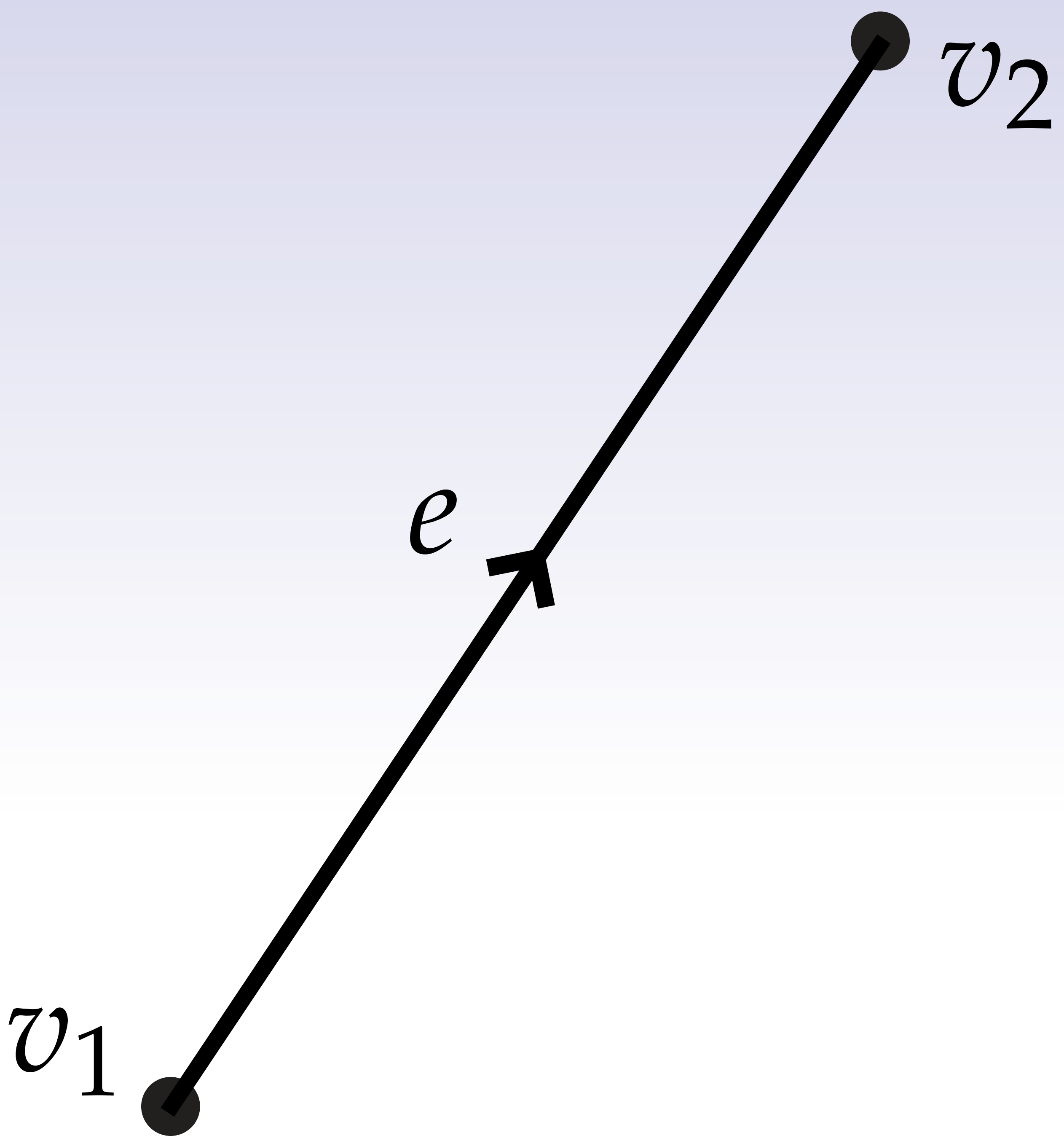


$$(\widehat{d\phi})_e = \int_e d\phi$$

Discrete Exterior Derivative (0-Forms)

ϕ - *primal 0-form* (vertices)

$d\phi$ - *primal 1-form* (edges)

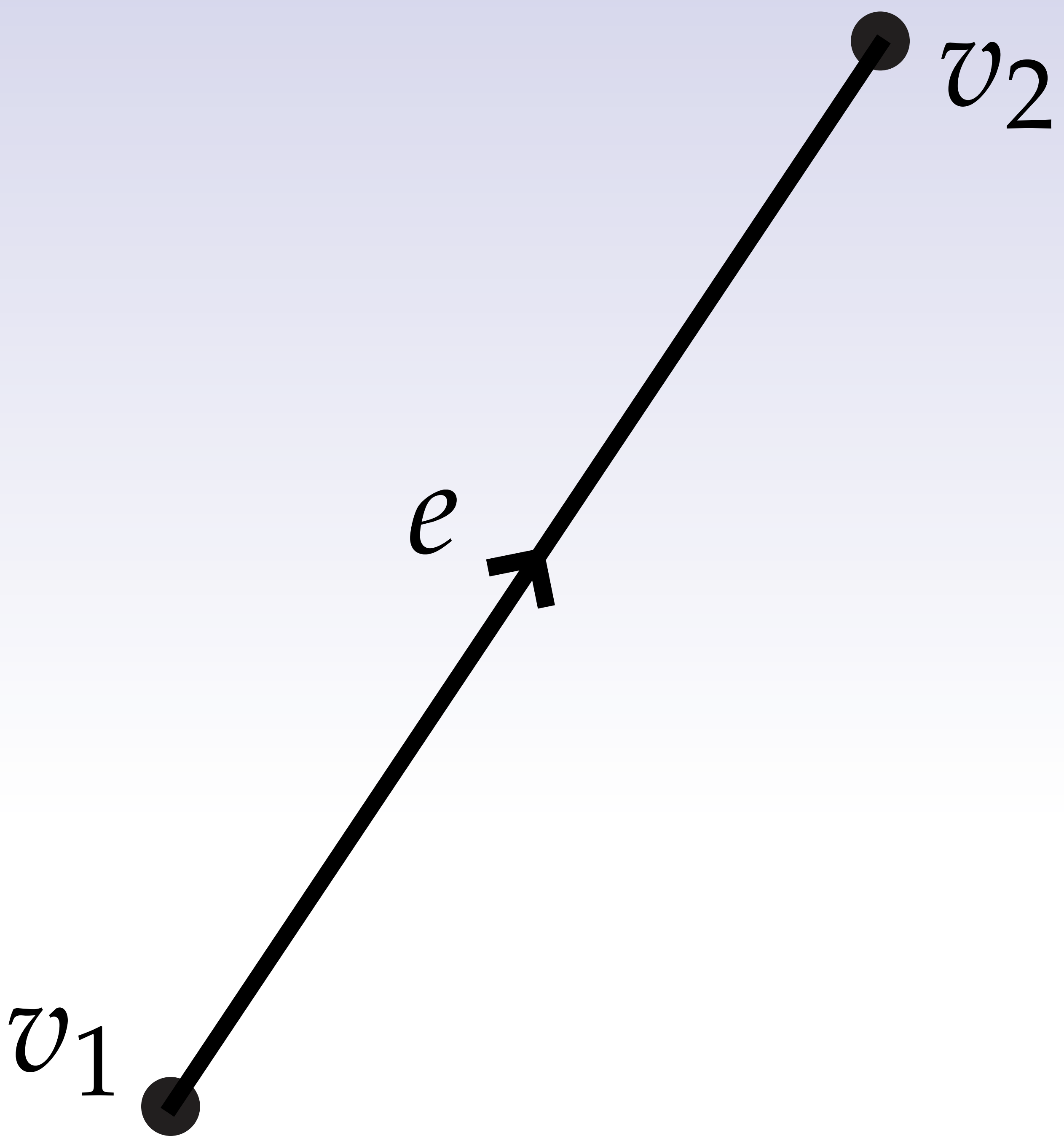


$$(\widehat{d\phi})_e = \int_e d\phi = \int_{\partial e} \phi$$

Discrete Exterior Derivative (0-Forms)

ϕ - *primal 0-form* (vertices)

$d\phi$ - *primal 1-form* (edges)

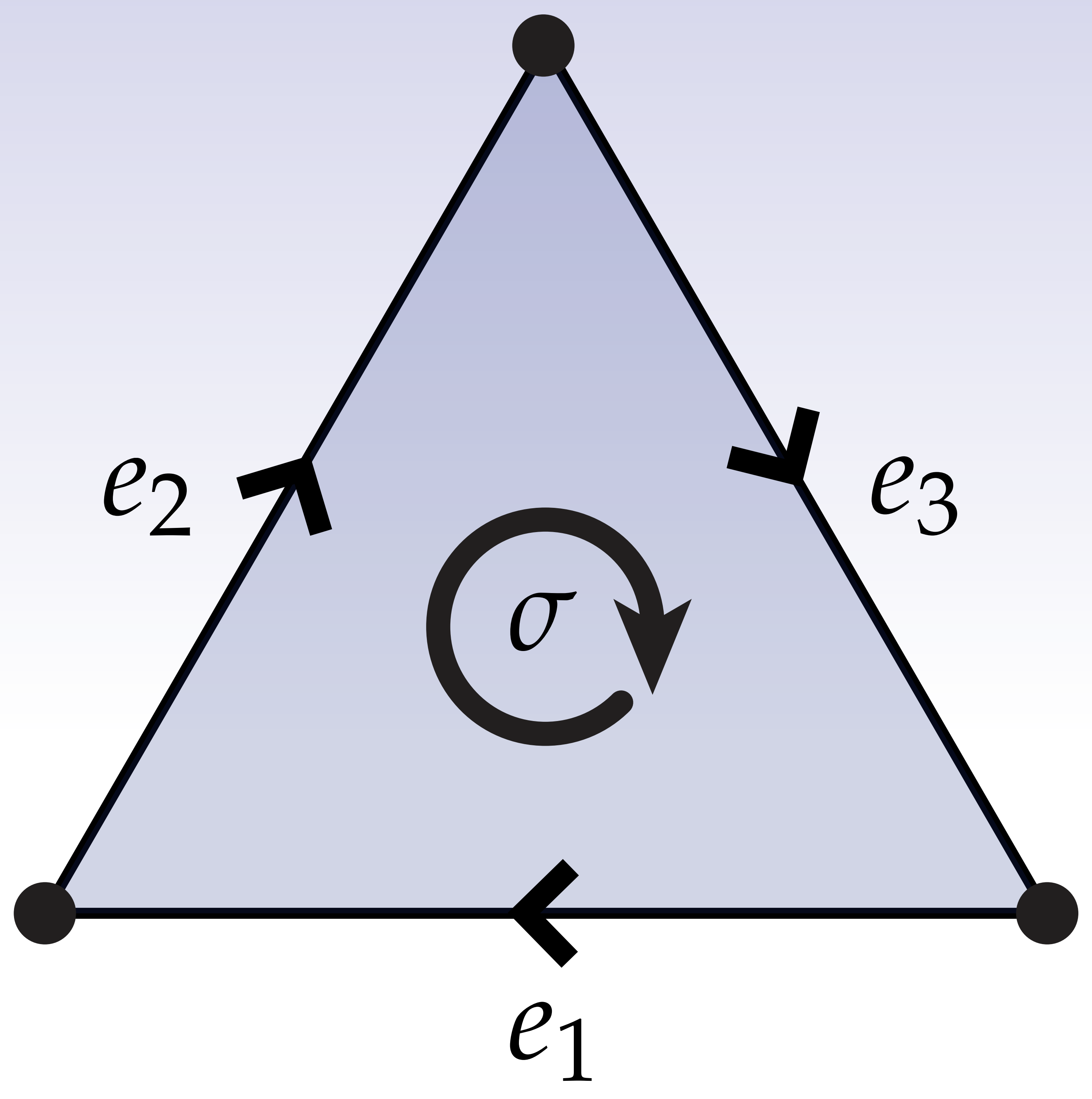


$$(\widehat{d\phi})_e = \int_e d\phi \; = \int_{\partial e} \phi \; = \hat{\phi}_2 - \hat{\phi}_1$$

Discrete Exterior Derivative (1-Forms)

α - *primal 1-form* (edges)

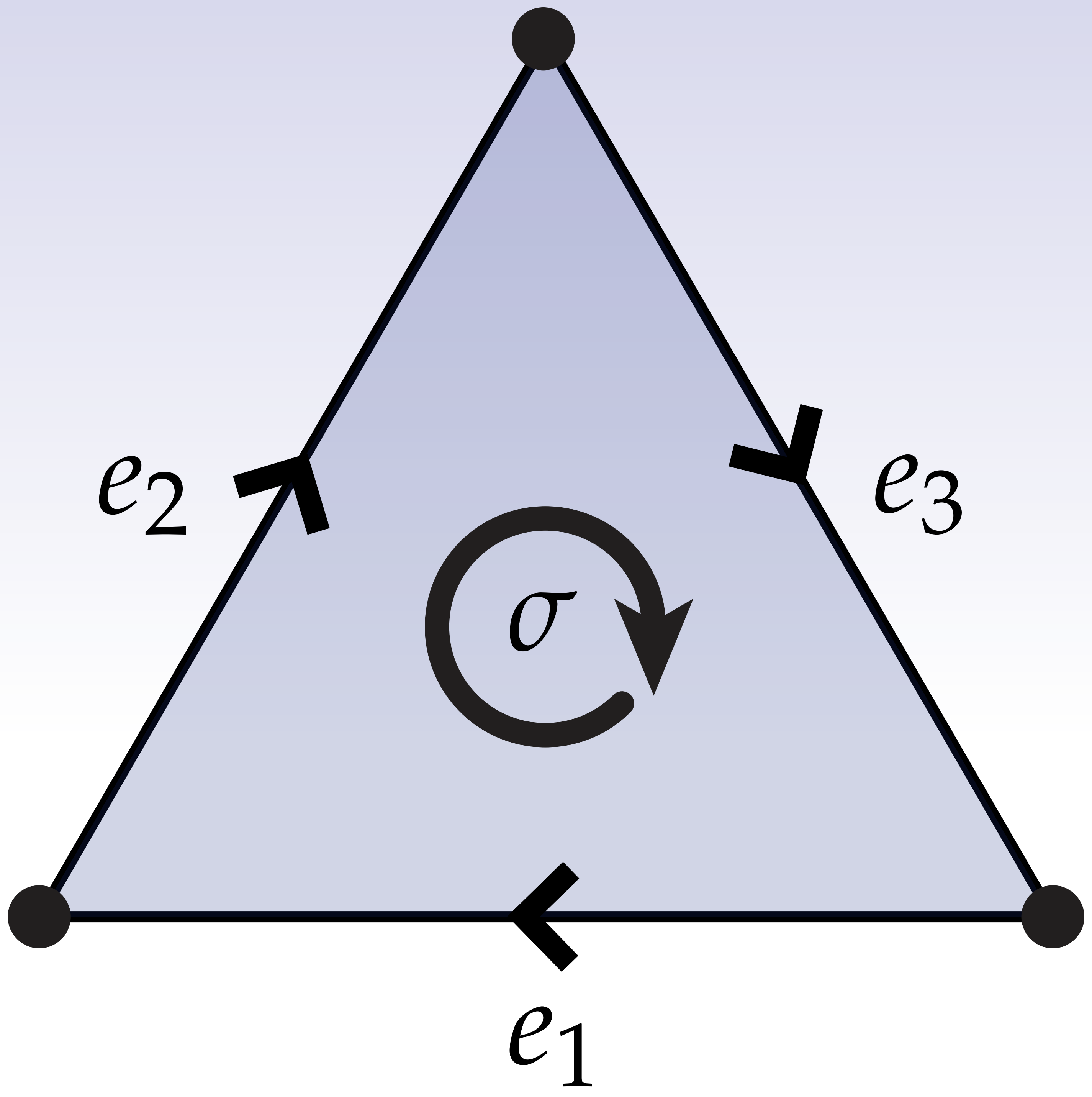
$d\alpha$ - *primal 2-form* (triangles)



Discrete Exterior Derivative (1-Forms)

α - *primal 1-form* (edges)

$d\alpha$ - *primal 2-form* (triangles)

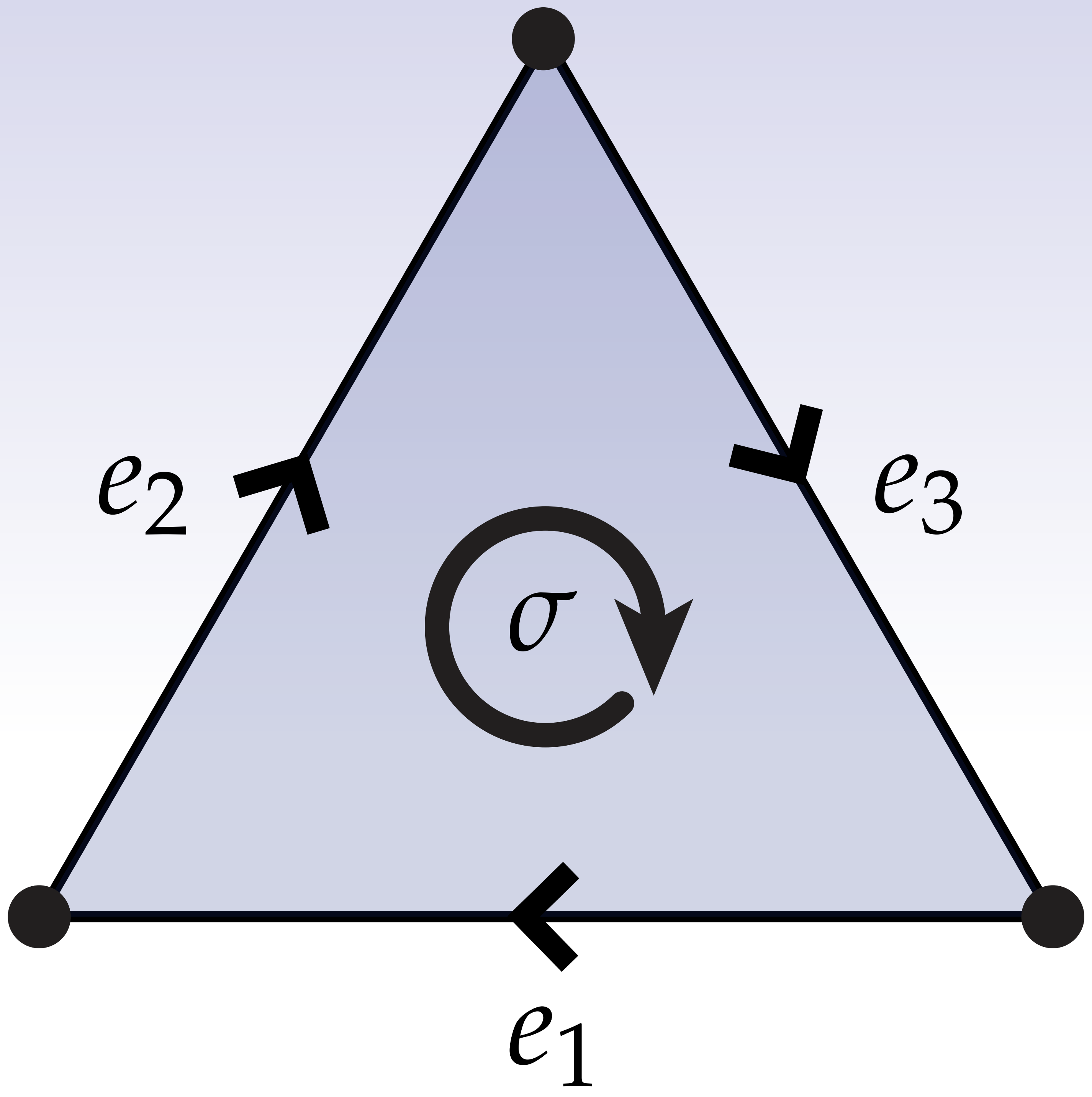


$$(\widehat{d\alpha})_\sigma$$

Discrete Exterior Derivative (1-Forms)

α - *primal 1-form* (edges)

$d\alpha$ - *primal 2-form* (triangles)

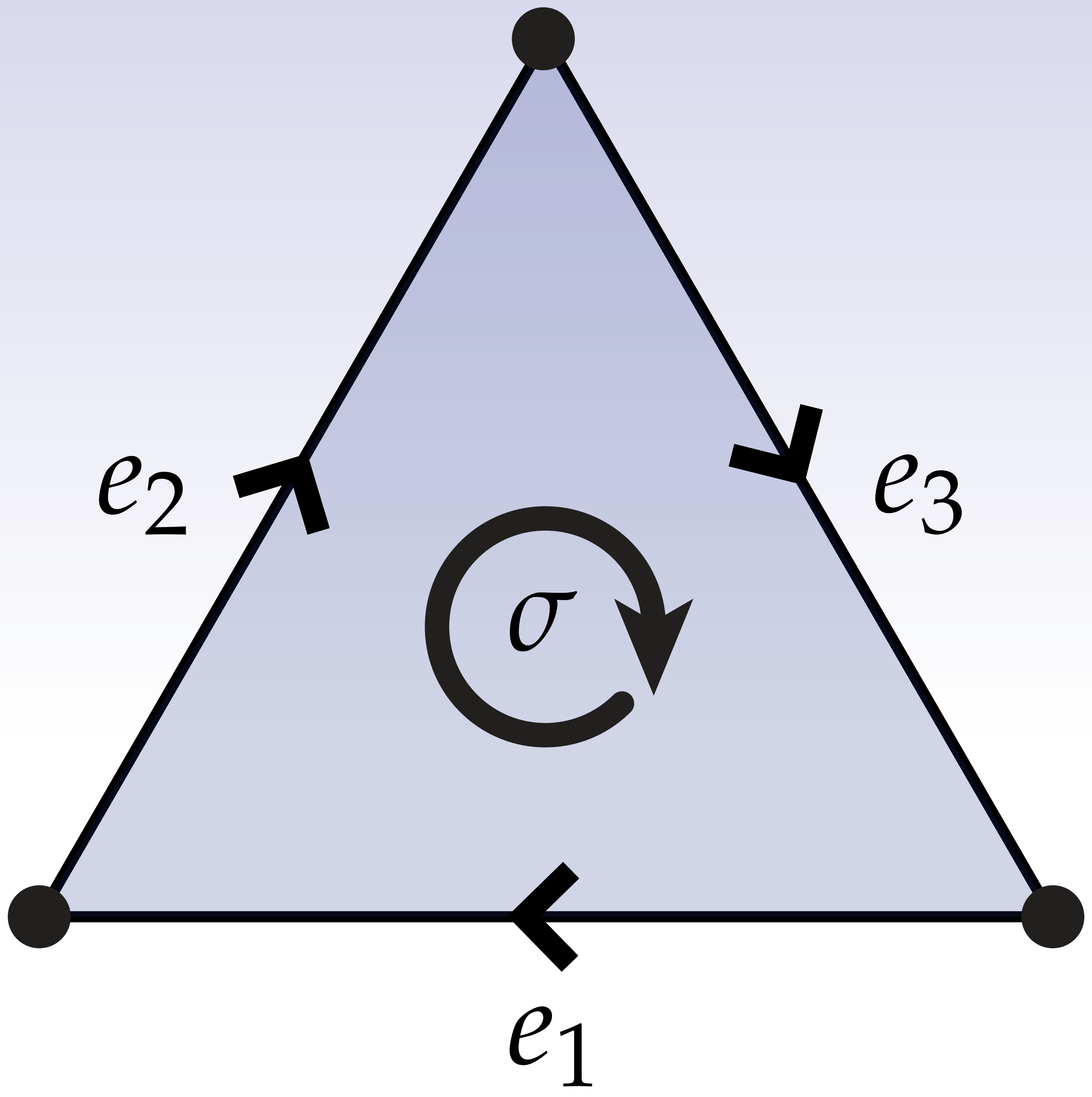


$$(\widehat{d\alpha})_\sigma = \int_\sigma d\alpha$$

Discrete Exterior Derivative (1-Forms)

α - *primal 1-form* (edges)

$d\alpha$ - *primal 2-form* (triangles)

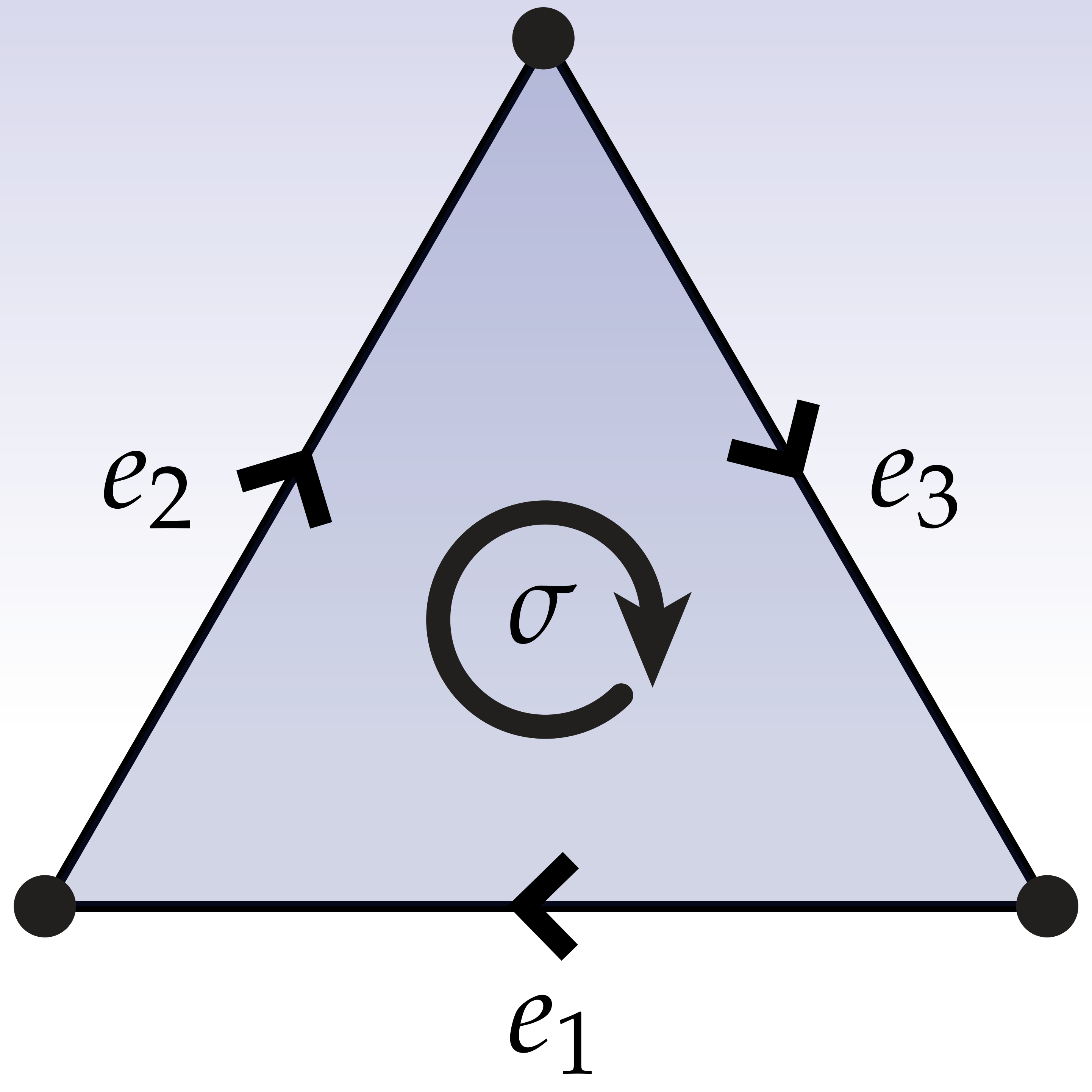


$$(\widehat{d\alpha})_\sigma = \int_\sigma d\alpha = \int_{\partial\sigma} \alpha$$

Discrete Exterior Derivative (1-Forms)

α - *primal 1-form* (edges)

$d\alpha$ - *primal 2-form* (triangles)

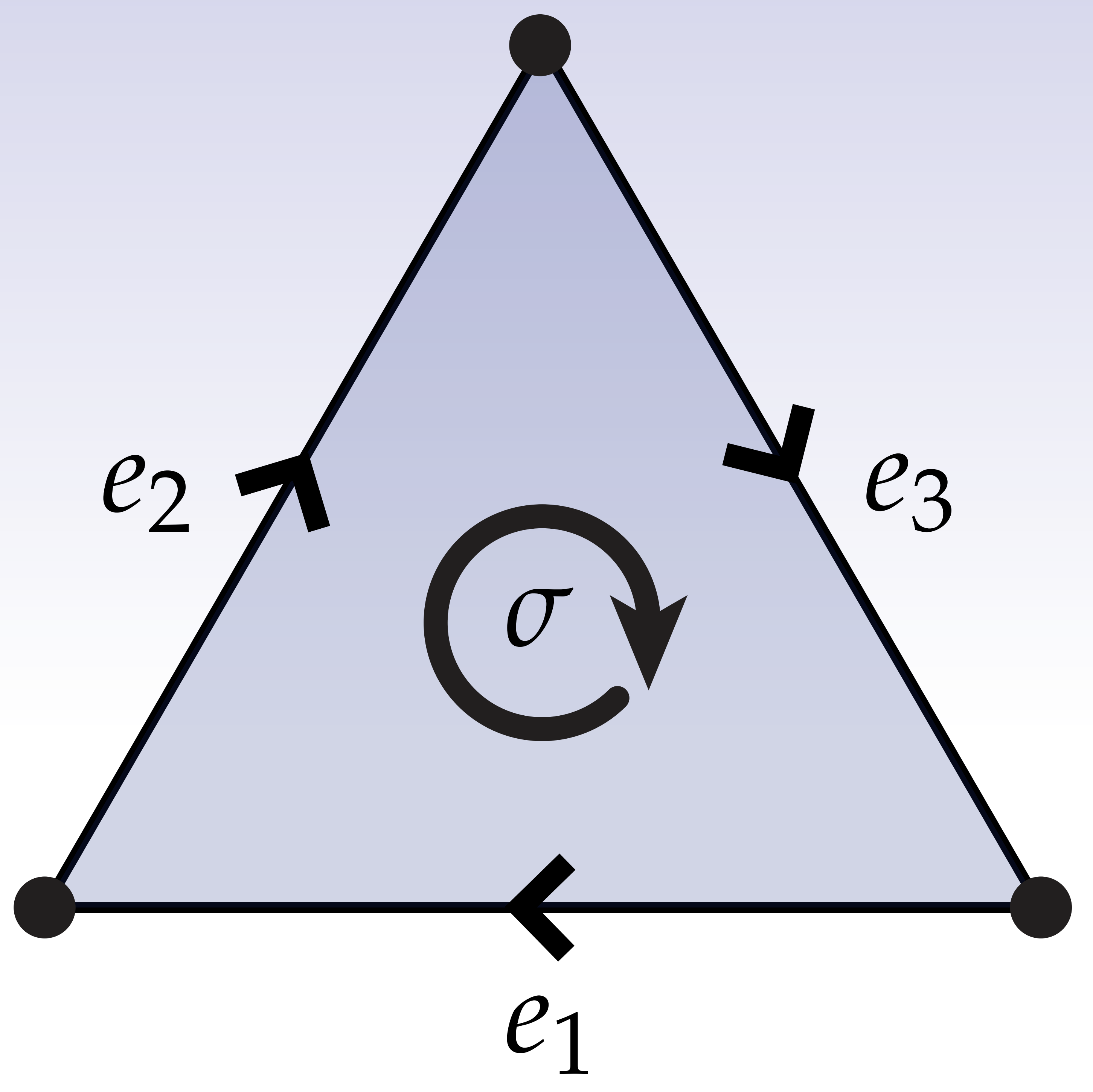


$$(\widehat{d\alpha})_\sigma = \int_\sigma d\alpha = \int_{\partial\sigma} \alpha = \sum_{i=1}^{3} \int_{e_i} \alpha$$

Discrete Exterior Derivative (1-Forms)

α - *primal 1-form* (edges)

$d\alpha$ - *primal 2-form* (triangles)



$$(\widehat{d\alpha})_\sigma = \int_\sigma d\alpha = \int_{\partial\sigma} \alpha = \sum_{i=1}^{3} \int_{e_i} \alpha = \sum_{i=1}^{3} \hat{\alpha}_i$$

Discrete Exterior Calculus - Summary

Discrete Exterior Calculus - Summary

- integrate k-form over k-simplices

Discrete Exterior Calculus - Summary

- integrate k-form over k-simplices
  - result is *discrete* k-form

Discrete Exterior Calculus - Summary

- integrate k-form over k-simplices
  - result is *discrete* k-form
  - sign changes according to orientation

Discrete Exterior Calculus - Summary

- integrate k-form over k-simplices
  - result is *discrete* k-form
  - sign changes according to orientation
- can also integrate over dual elements (*dual* forms)

Discrete Exterior Calculus - Summary

- integrate *k*-form over *k*-simplices
  - result is *discrete k*-form
  - sign changes according to orientation
- can also integrate over dual elements (*dual* forms)
- Hodge star converts between primal and dual (*approximately*!)

Discrete Exterior Calculus - Summary

- integrate *k*-form over *k*-simplices
  - result is *discrete k*-form
  - sign changes according to orientation
- can also integrate over dual elements (*dual* forms)
- Hodge star converts between primal and dual (*approximately*!)
  - multiply by ratio of dual/primal volume

Discrete Exterior Calculus - Summary

- integrate k-form over k-simplices
  - result is *discrete* k-form
  - sign changes according to orientation
- can also integrate over dual elements (*dual* forms)
- Hodge star converts between primal and dual (*approximately*!)
  - multiply by ratio of dual/primal volume
- discrete exterior derivative is just a sum

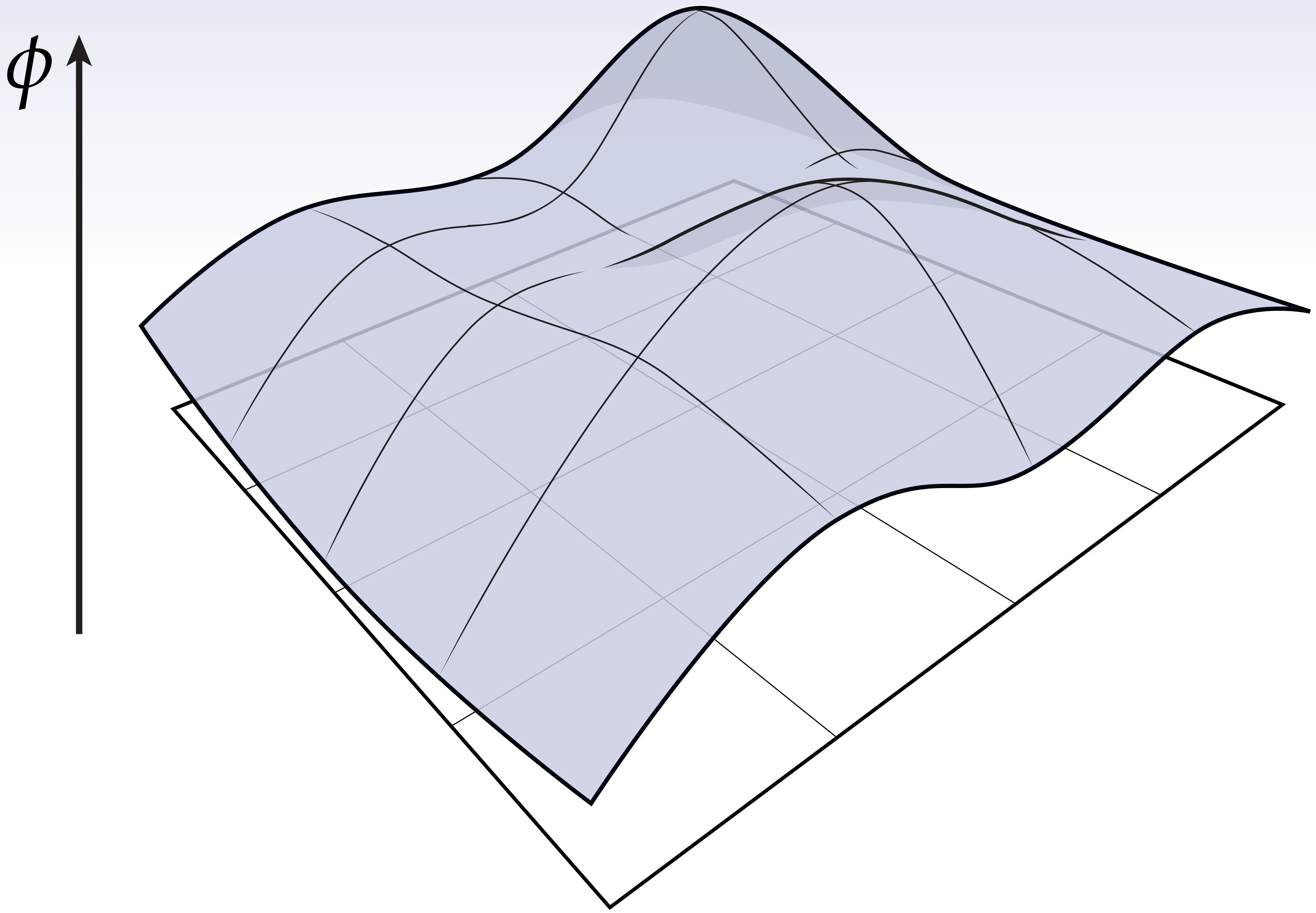
Discrete Exterior Calculus - Summary

- integrate *k*-form over *k*-simplices
  - result is *discrete k*-form
  - sign changes according to orientation
- can also integrate over dual elements (*dual* forms)
- Hodge star converts between primal and dual (*approximately*!)
  - multiply by ratio of dual/primal volume
- discrete exterior derivative is just a sum
  - gives *exact* value (via Stokes' theorem)

The Poisson Equation

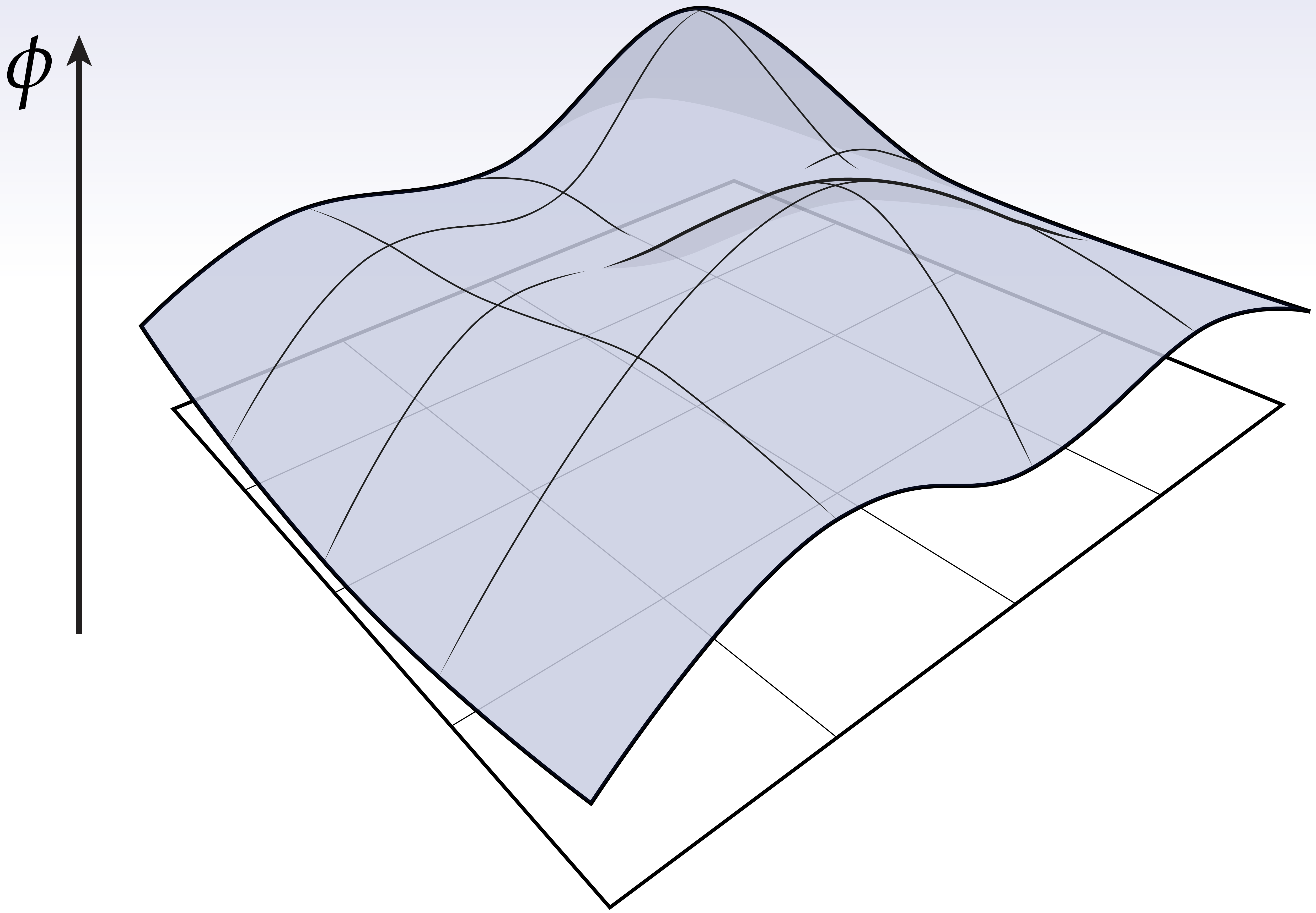
Laplacian

Laplacian



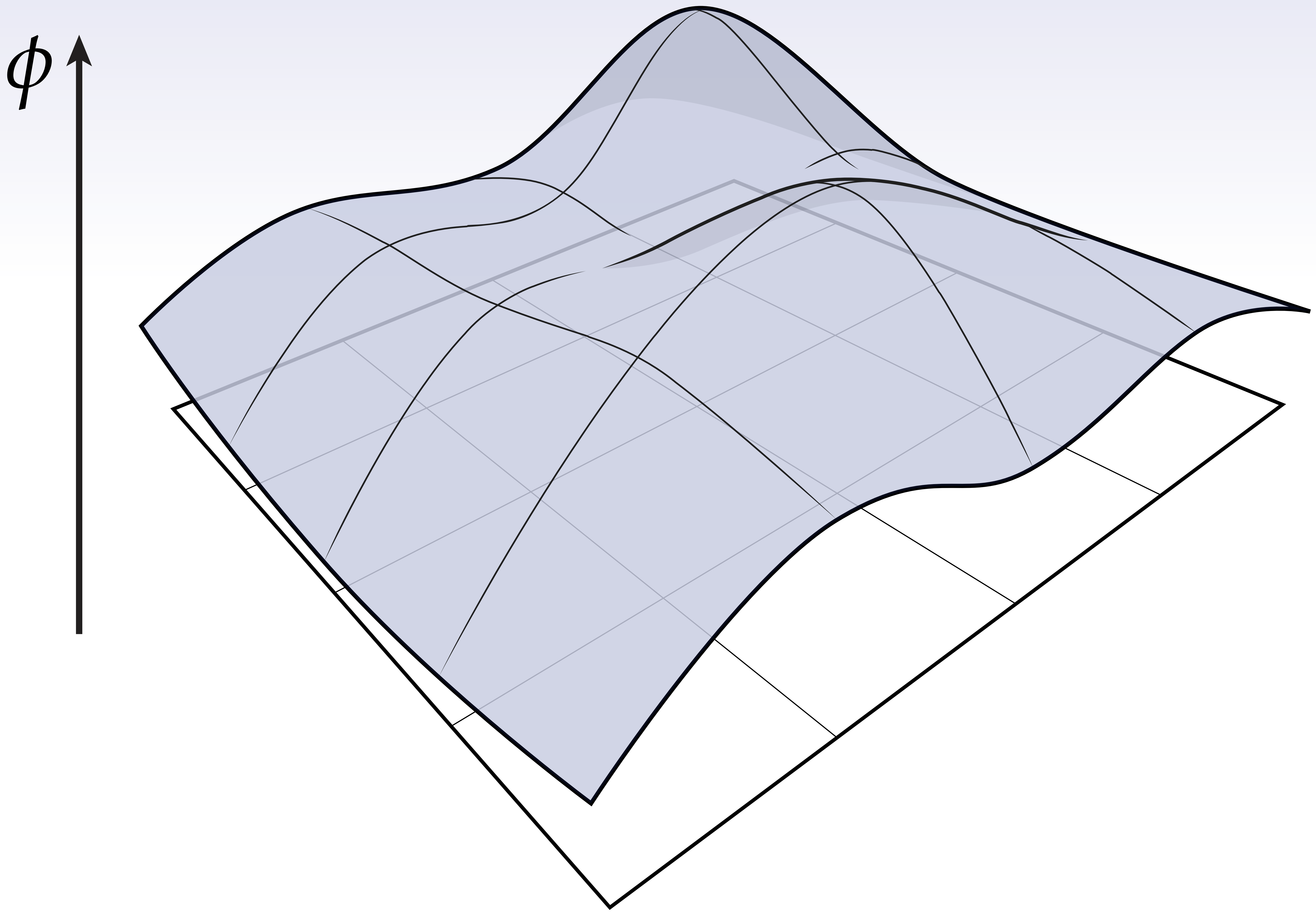
Laplacian

$\Delta\phi$



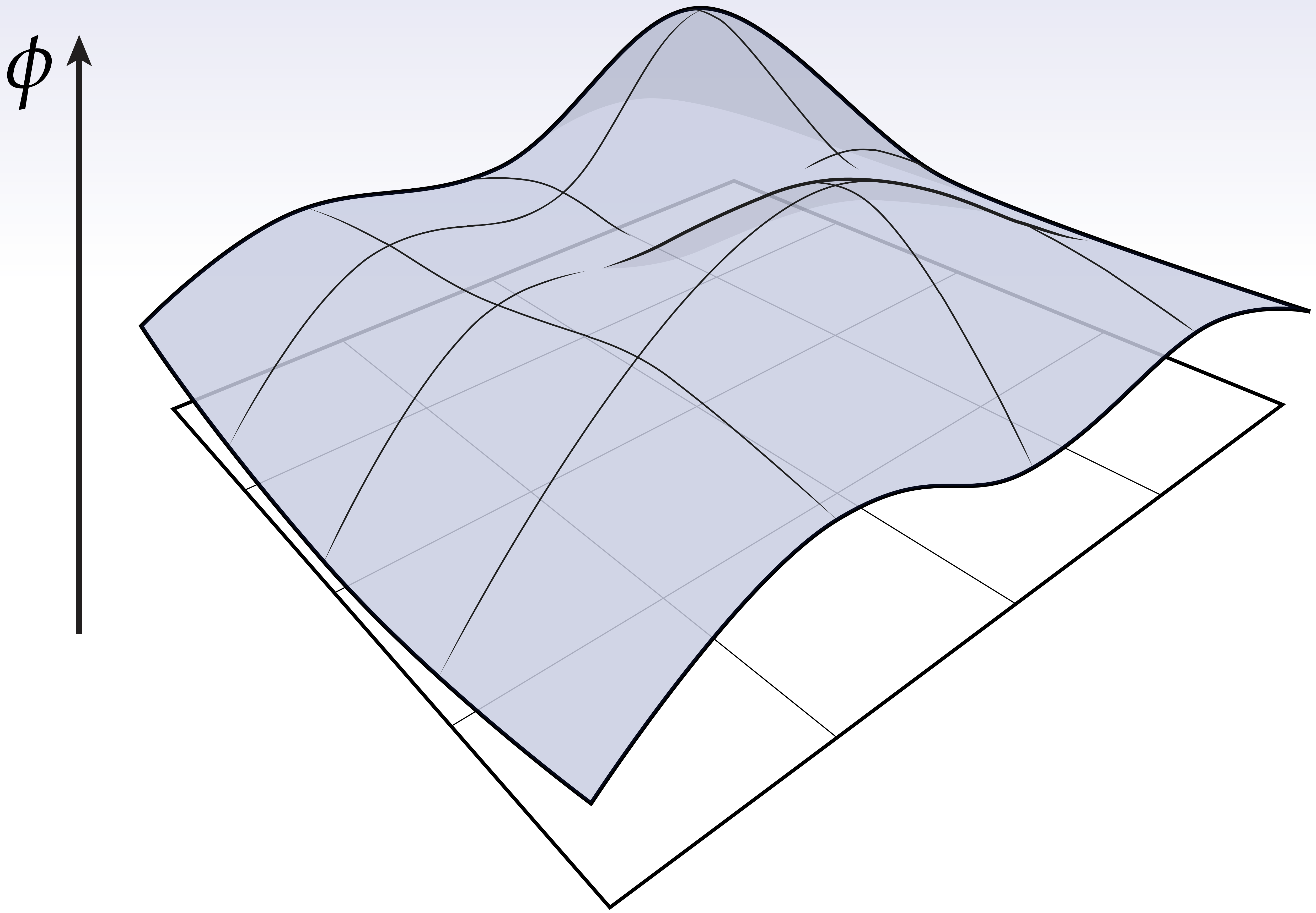
Laplacian

$$\Delta\phi = \nabla \cdot \nabla\phi$$



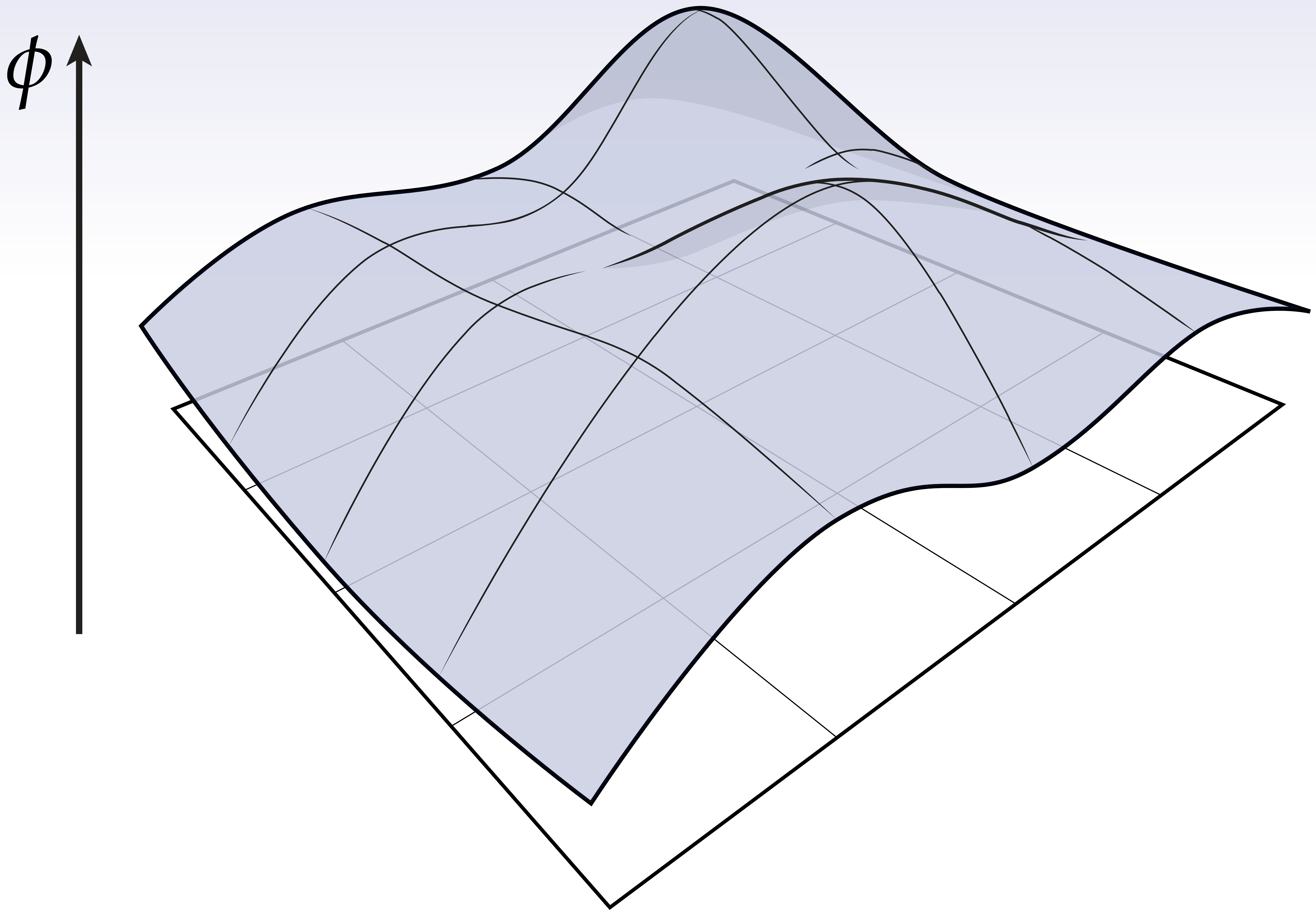
Laplacian

$$\Delta\phi = \nabla \cdot \nabla\phi = \star d \star d\, \phi$$



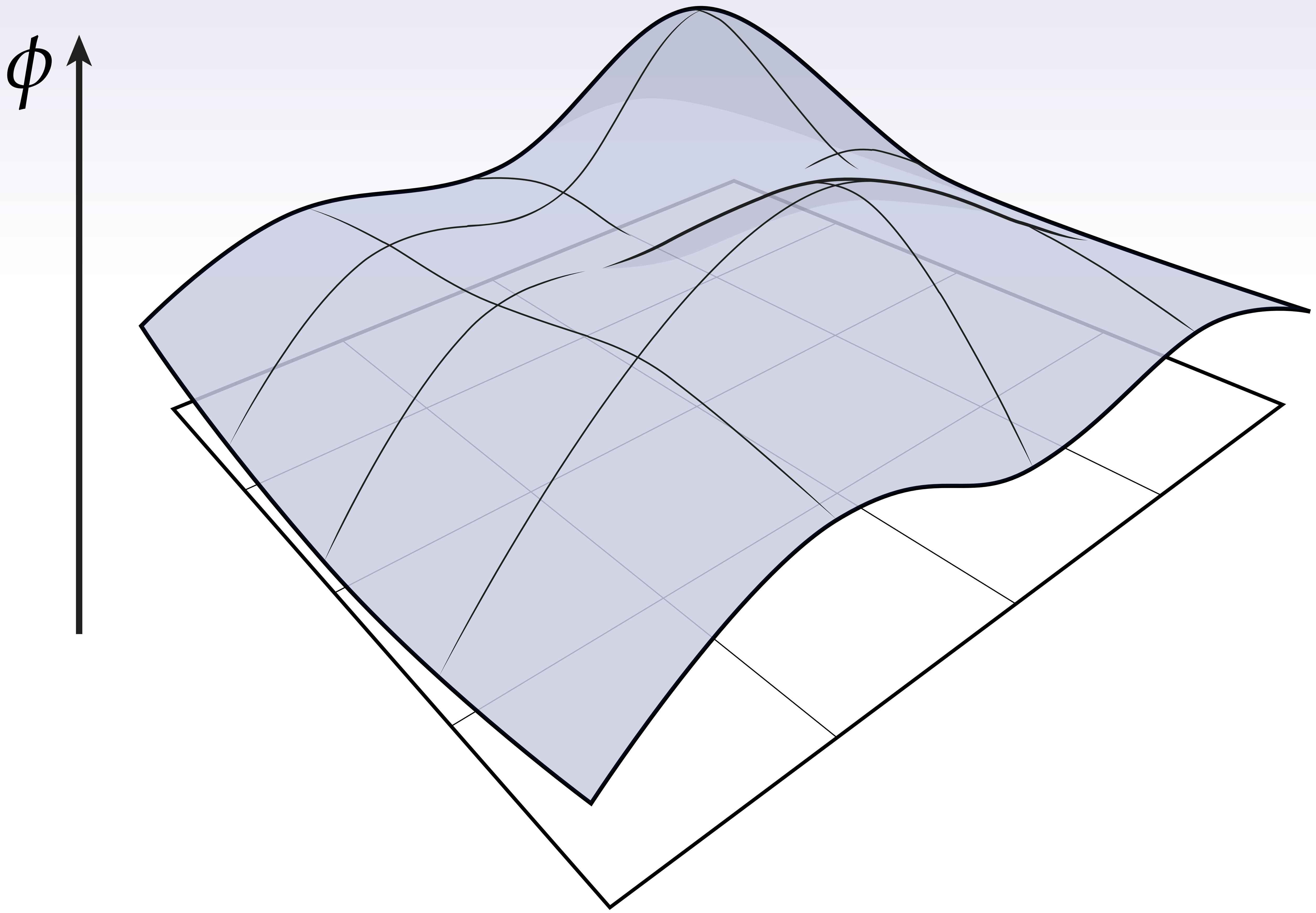
Laplacian

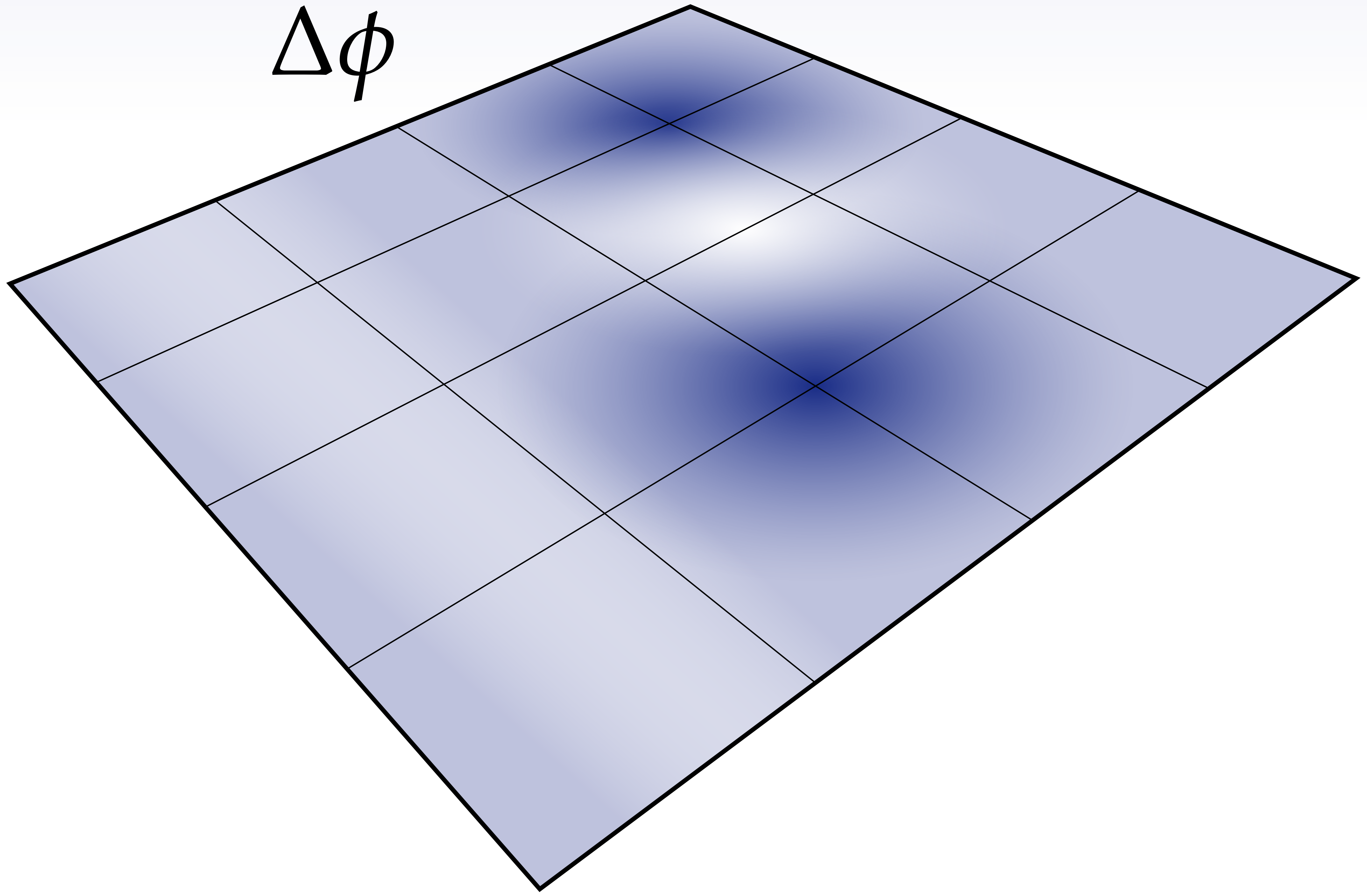
$$\Delta\phi = \nabla \cdot \nabla\phi = \star d \star d\,\phi = \delta d\phi$$



Laplacian

$$\Delta\phi = \nabla \cdot \nabla\phi = \star d \star d\,\phi = \delta d\phi$$



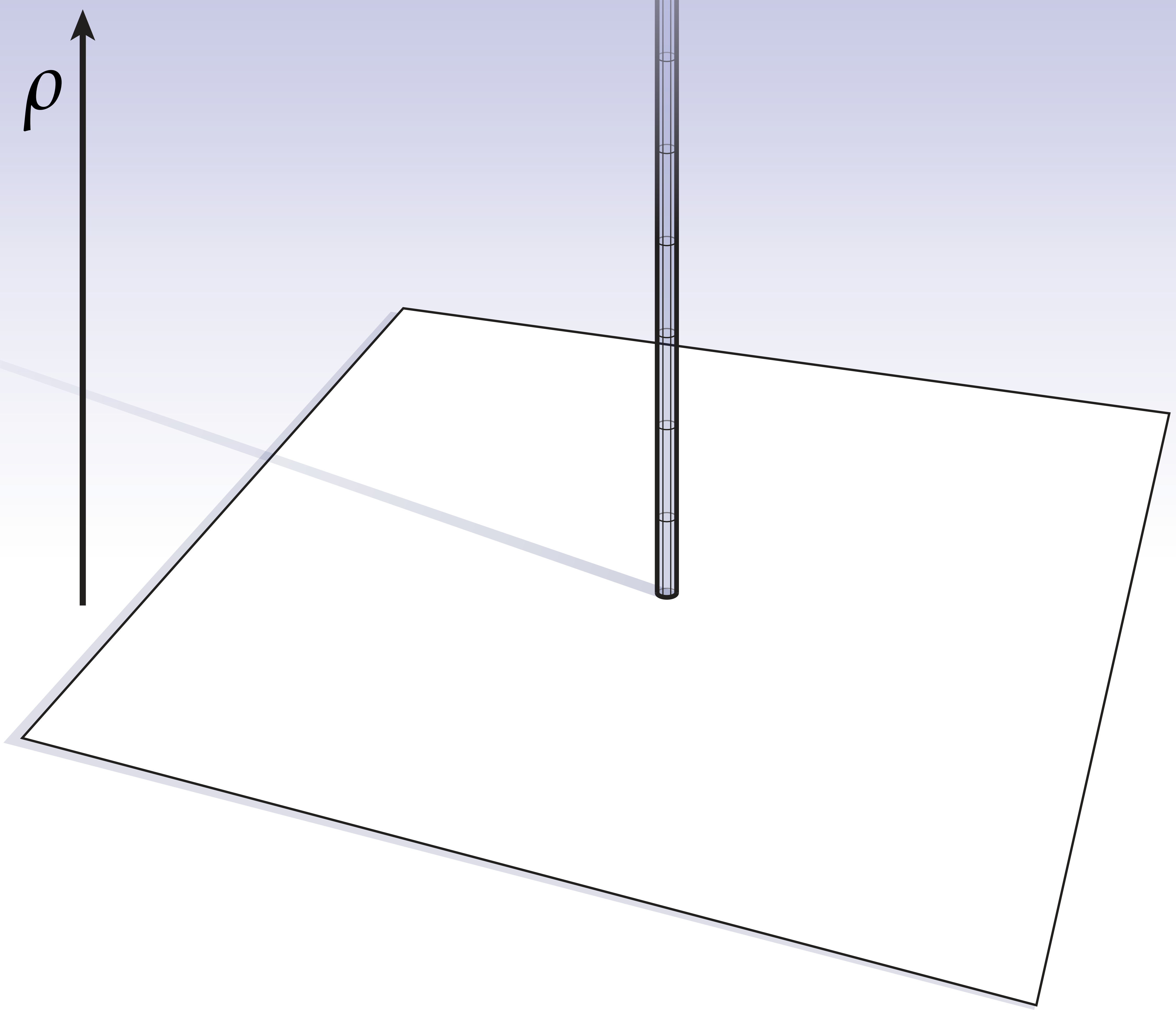


Poisson Equation

Poisson Equation

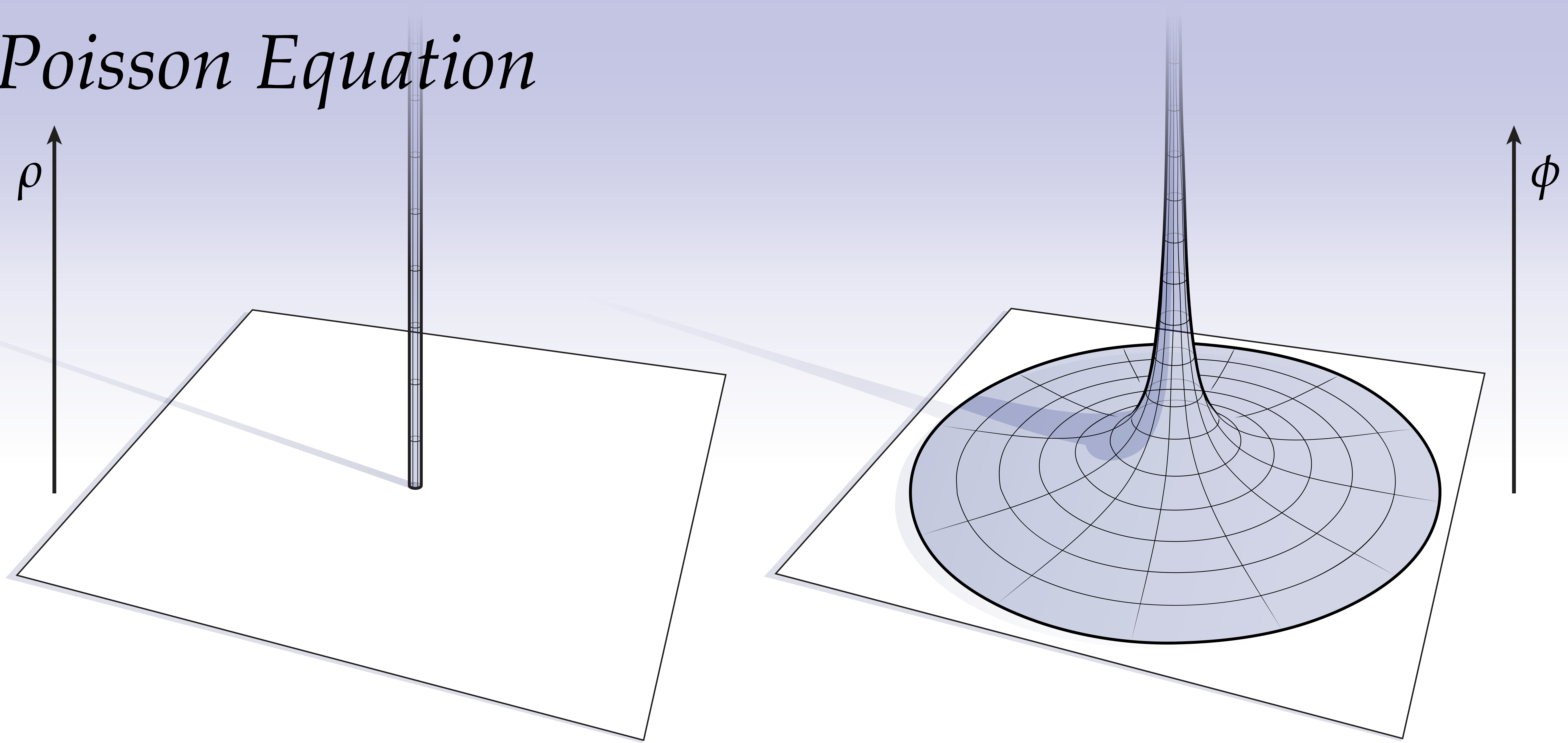
$$\Delta\phi = \rho$$

Poisson Equation



$$\Delta\phi = \rho$$

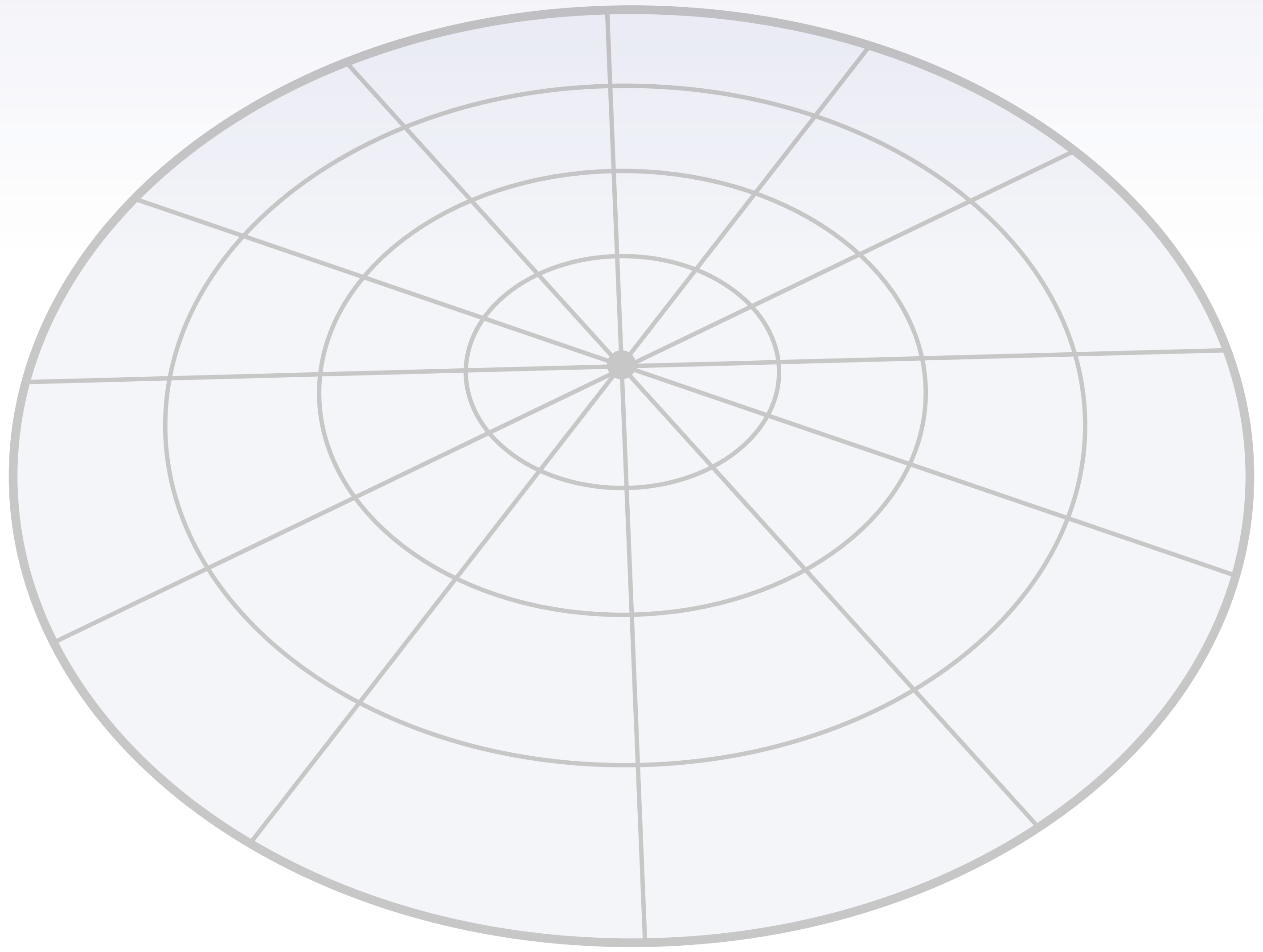
Poisson Equation



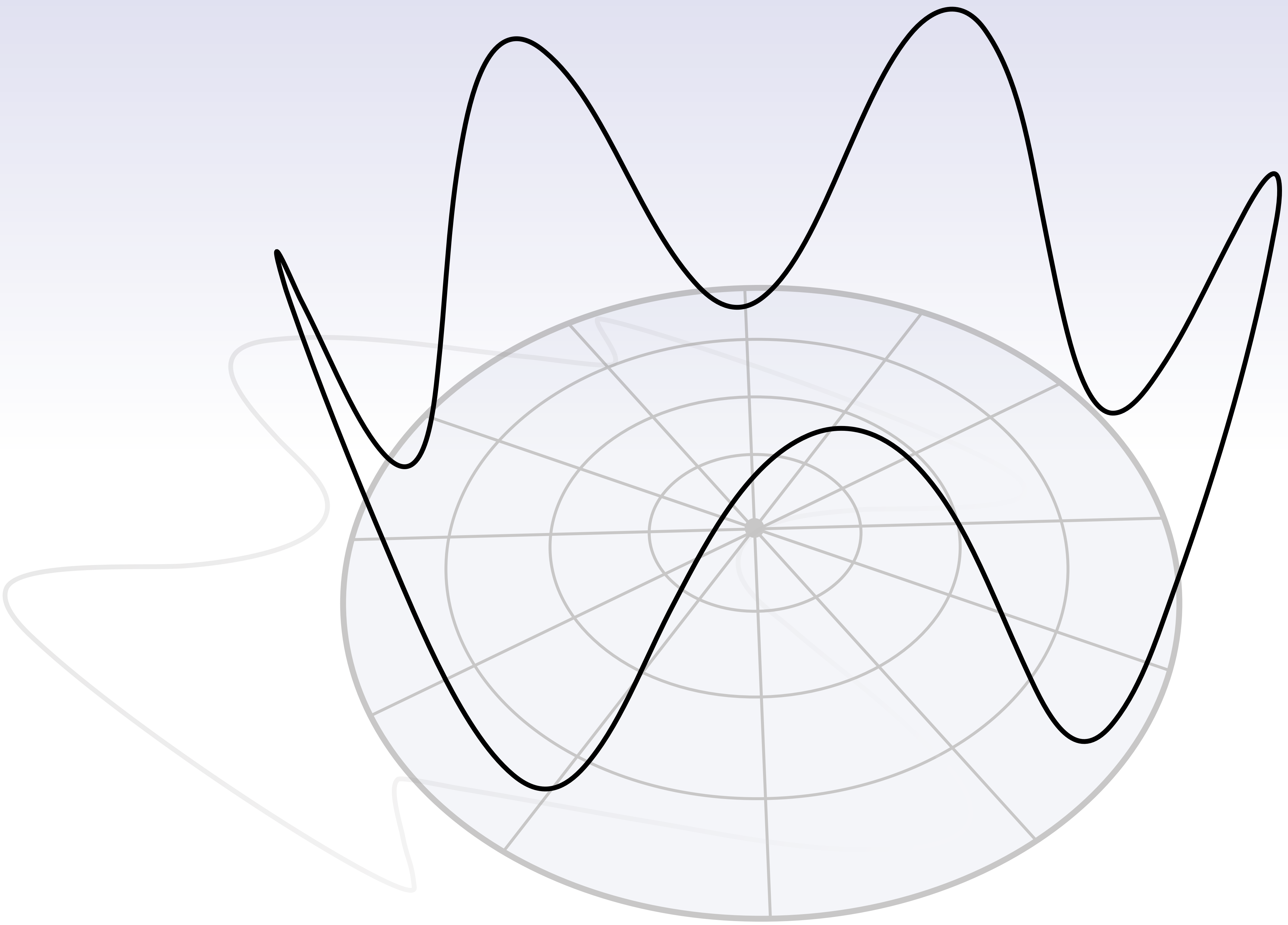
$$\Delta\phi = \rho$$

Harmonic Functions

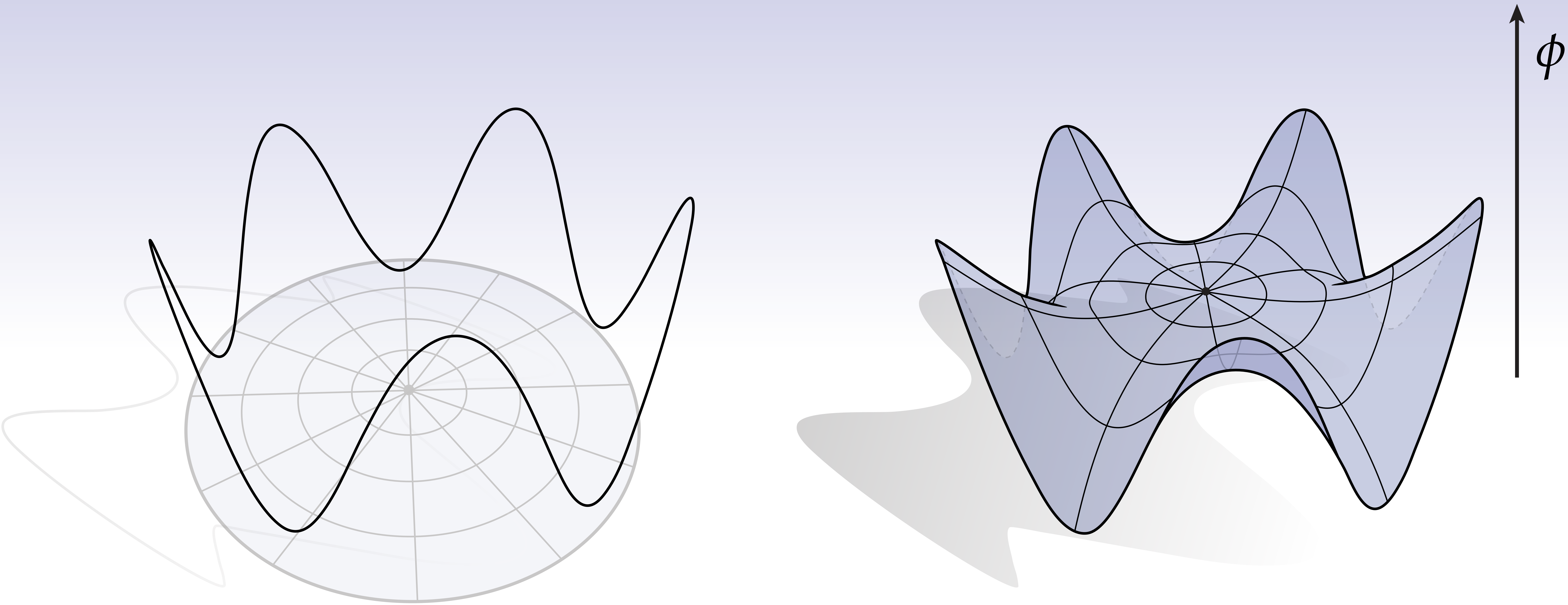
Harmonic Functions



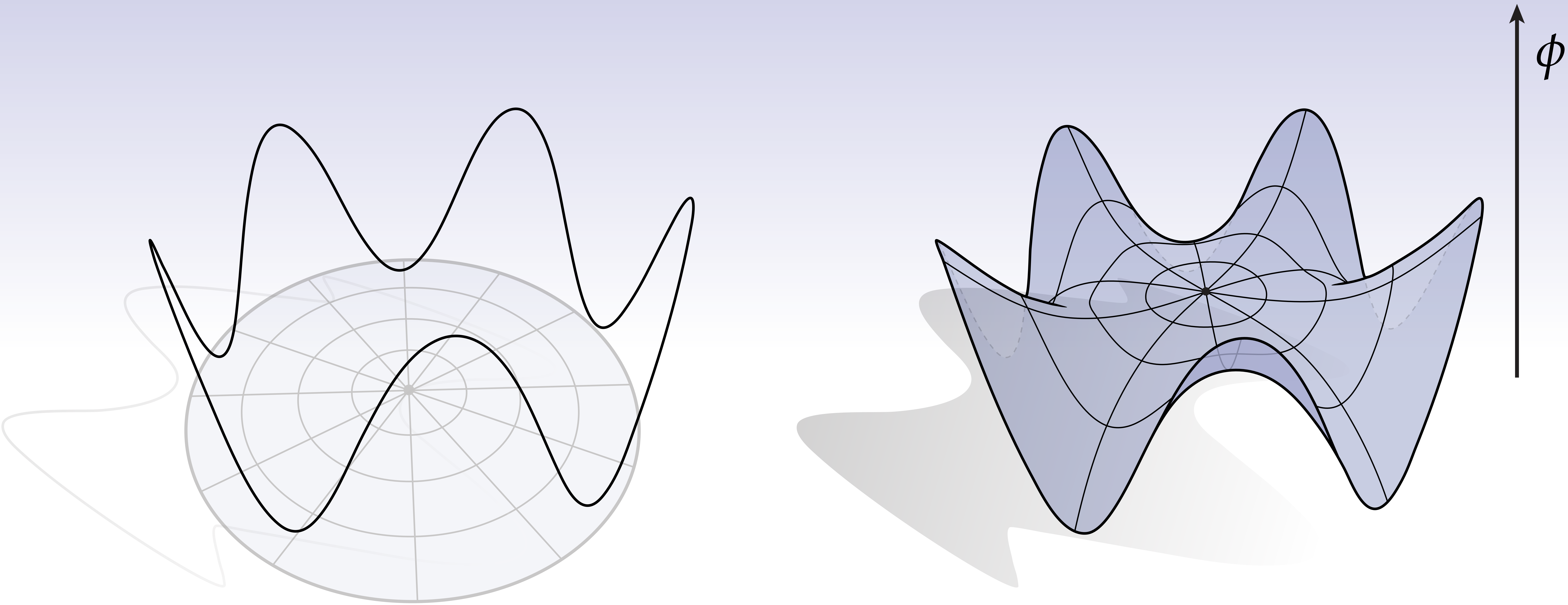
Harmonic Functions



Harmonic Functions



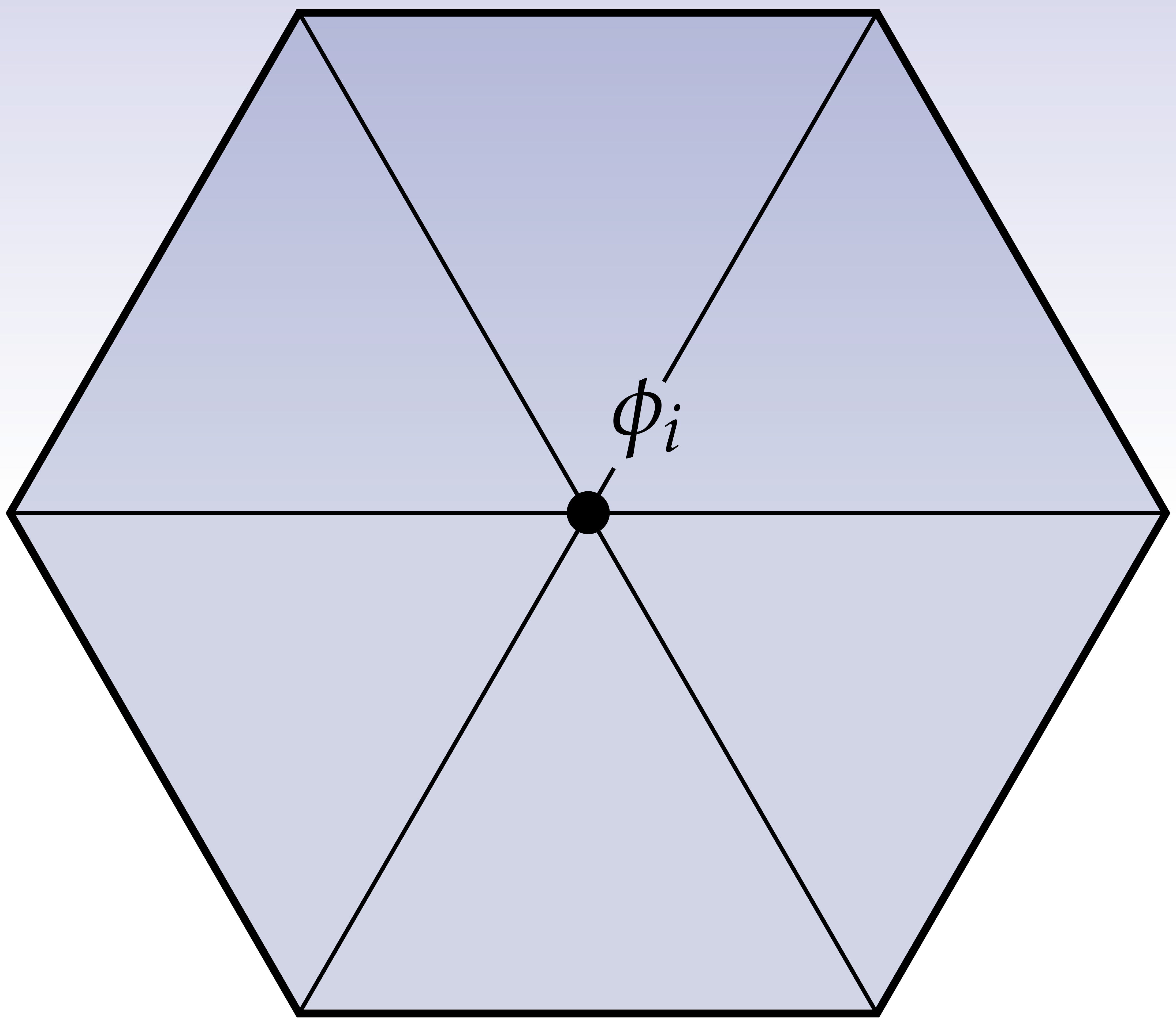
Harmonic Functions



$$\Delta\phi = 0$$

Discrete Laplacian

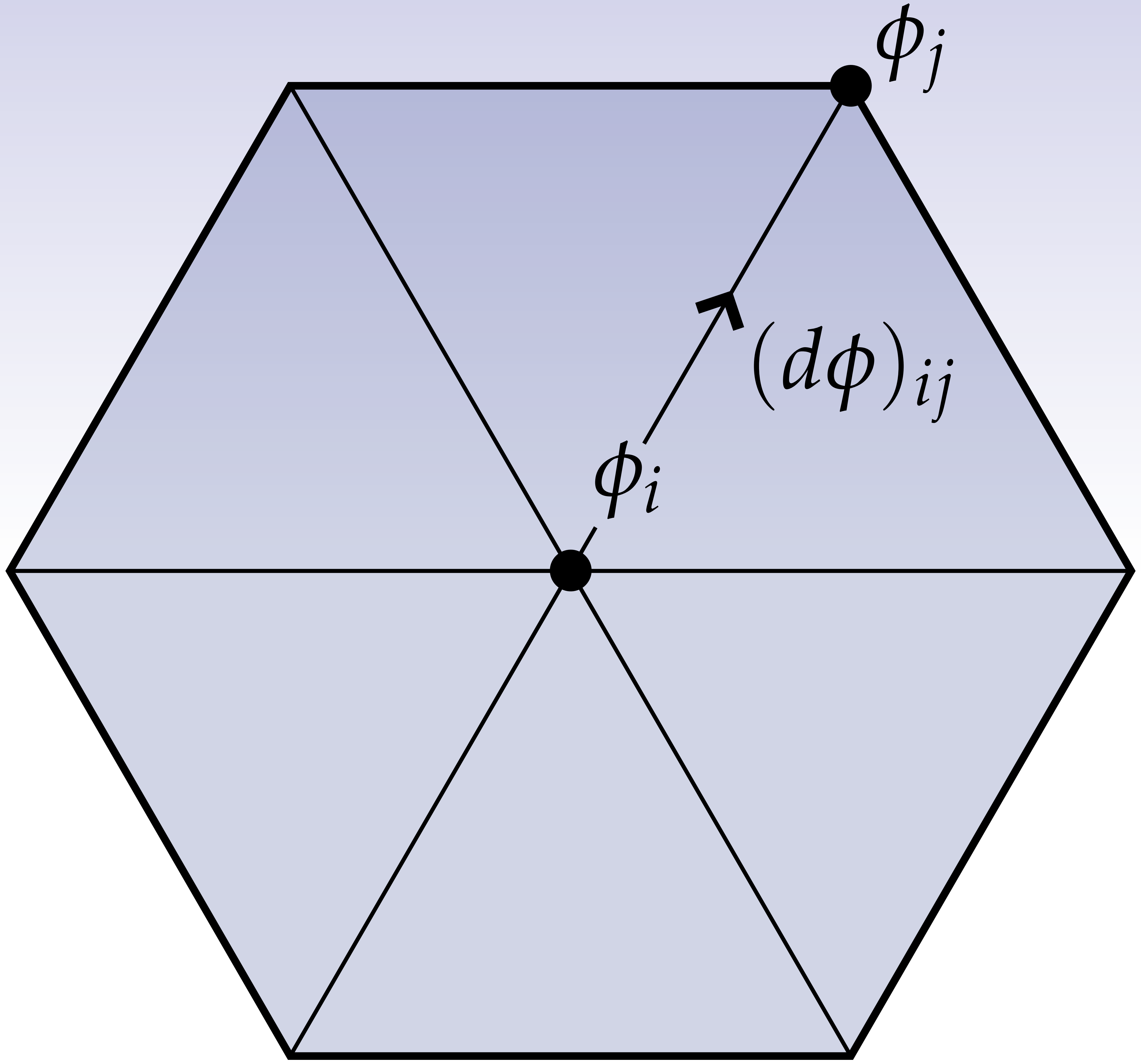
$$\boxed{\Delta\phi = \star d \star d\phi}$$



Discrete Laplacian

$$\boxed{\Delta\phi = \star d \star d\phi}$$

$$(d\phi)_{ij} = \phi_j - \phi_i$$

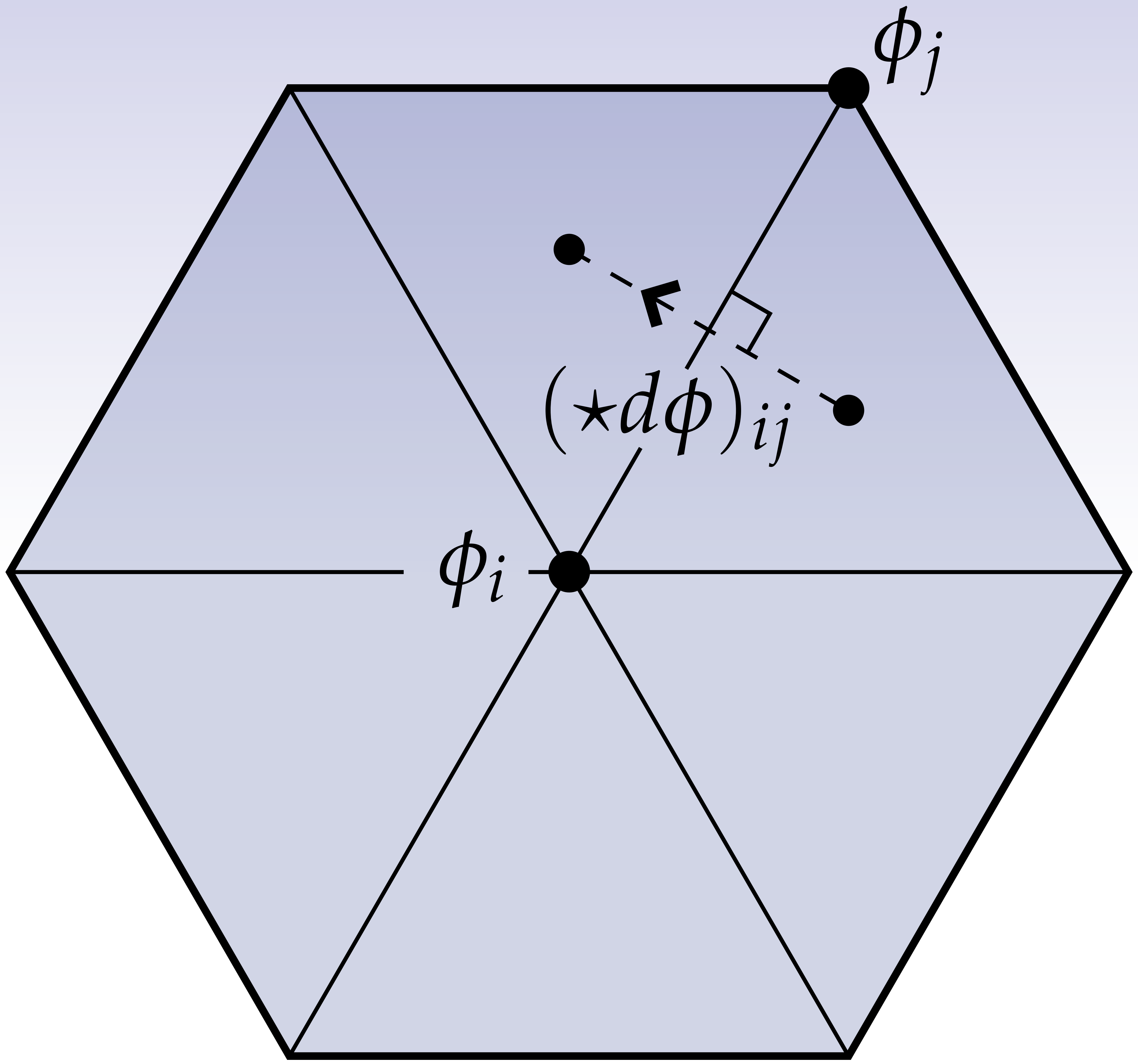


Discrete Laplacian

$$\boxed{\Delta\phi = \star d \star d\phi}$$

$$(d\phi)_{ij} = \phi_j - \phi_i$$

$$(\star d\phi)_{ij} = \frac{\ell^\star_{ij}}{\ell_{ij}}(\phi_j - \phi_i)$$



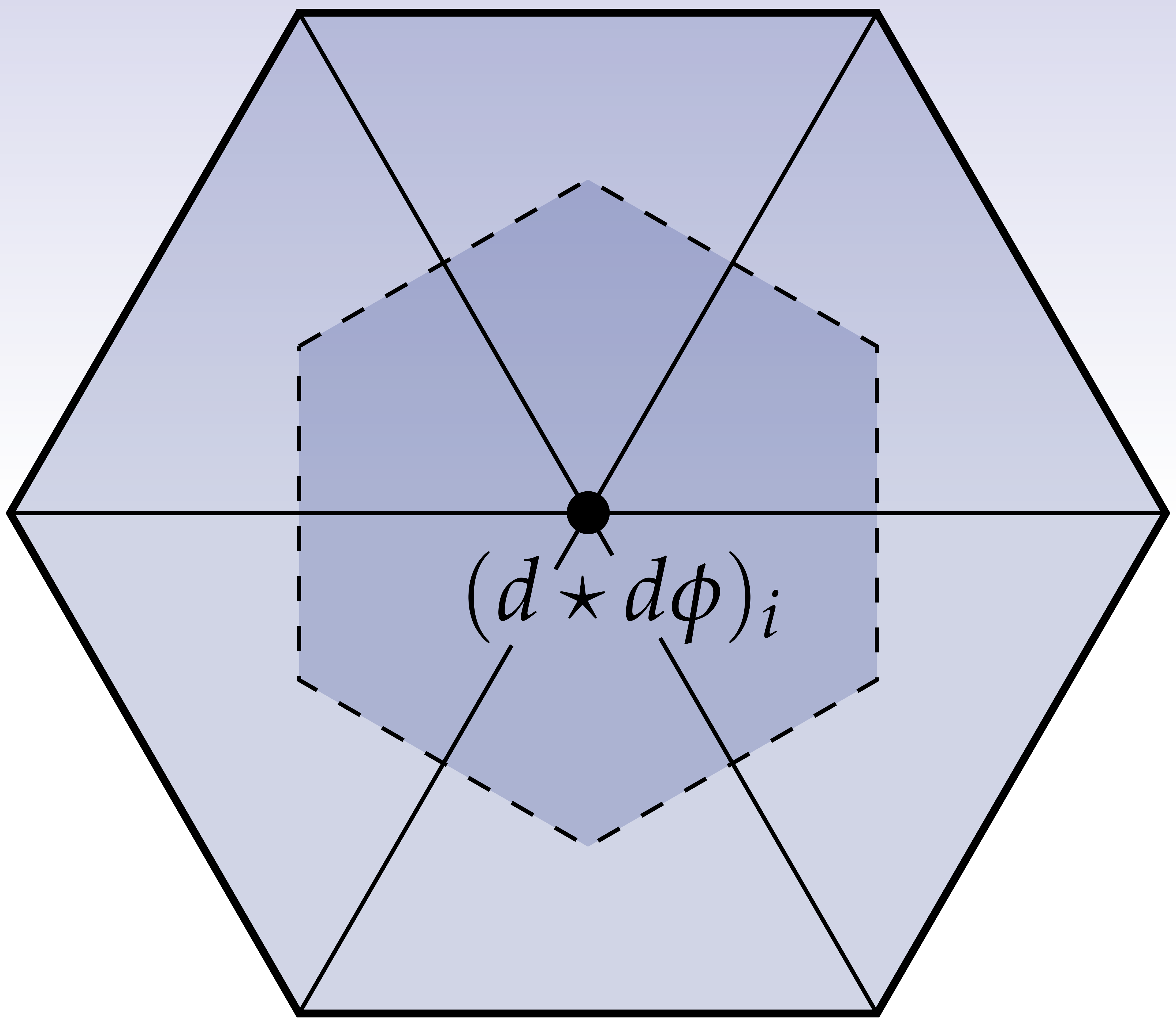
Discrete Laplacian

$$\boxed{\Delta\phi = \star d \star d\phi}$$

$$(d\phi)_{ij} = \phi_j - \phi_i$$

$$(\star d\phi)_{ij} = \frac{\ell^\star_{ij}}{\ell_{ij}}(\phi_j - \phi_i)$$

$$(d \star d\phi)_i = \sum_j \frac{\ell^\star_{ij}}{\ell_{ij}}(\phi_j - \phi_i)$$



Discrete Laplacian

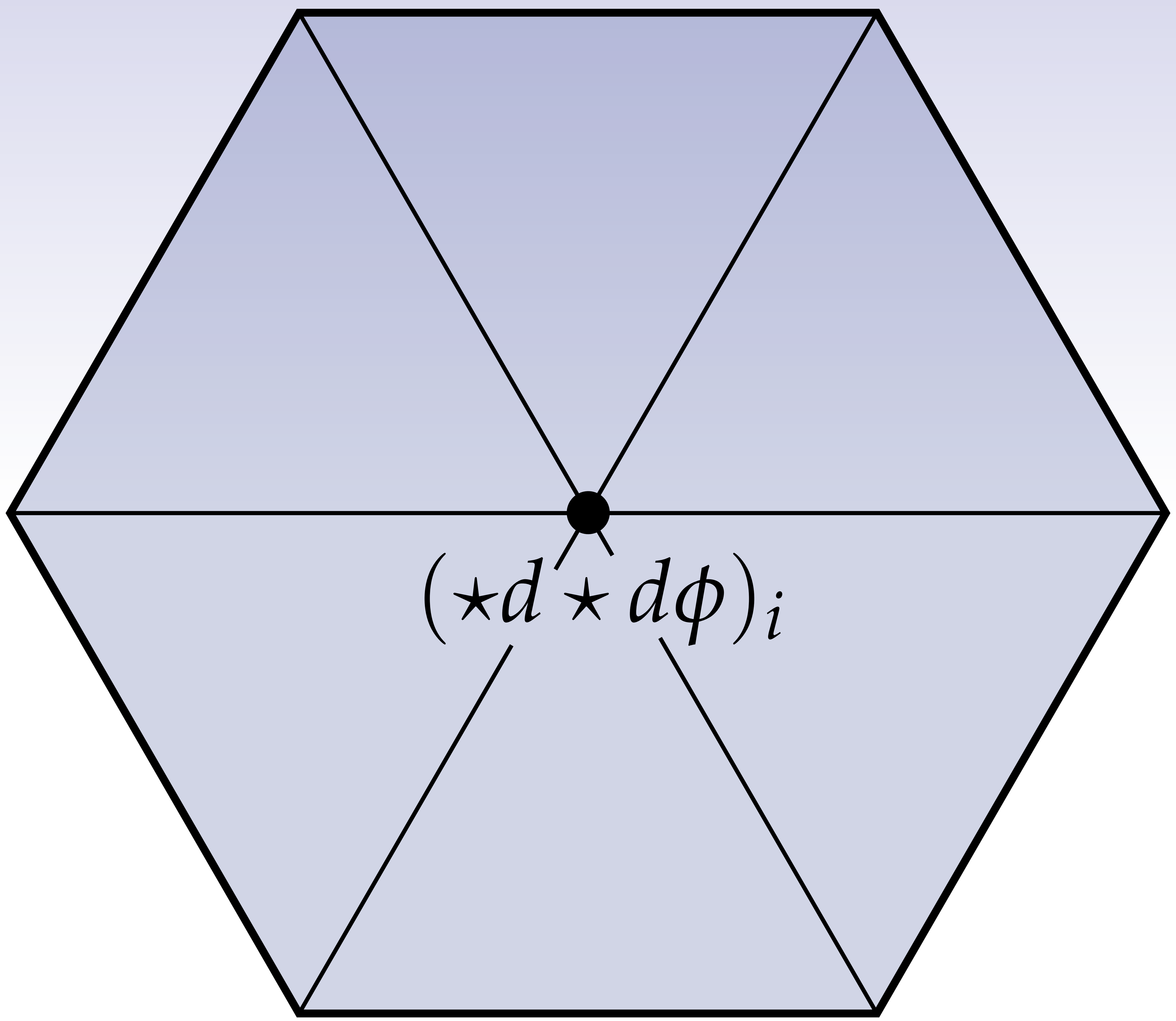
$$\boxed{\Delta\phi = \star d \star d\phi}$$

$$(d\phi)_{ij} = \phi_j - \phi_i$$

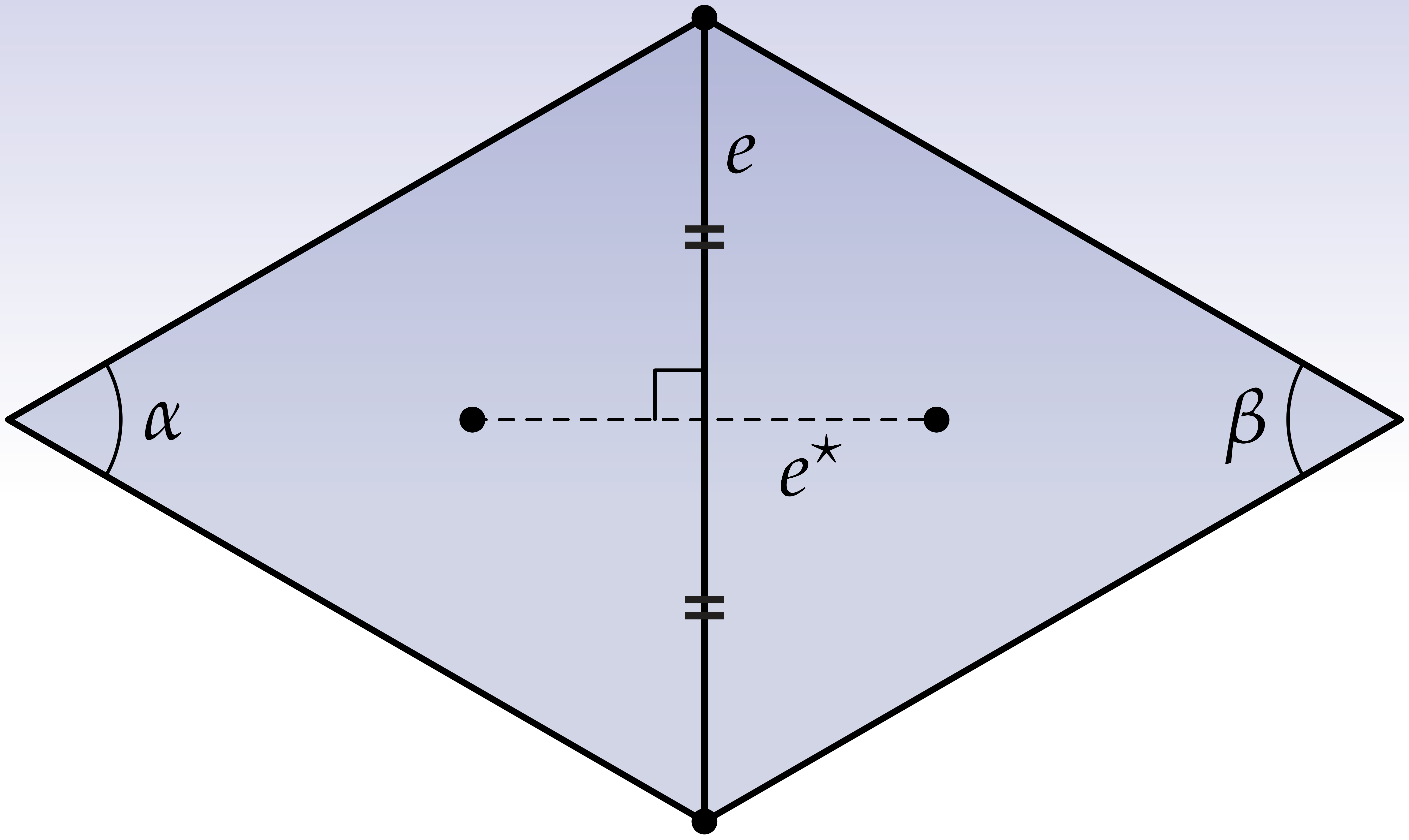
$$(\star d\phi)_{ij} = \frac{\ell^\star_{ij}}{\ell_{ij}}(\phi_j - \phi_i)$$

$$(d \star d\phi)_i = \sum_j \frac{\ell^\star_{ij}}{\ell_{ij}}(\phi_j - \phi_i)$$

$$(\star d \star d\phi)_i = \frac{1}{\mathcal{V}_i}\sum_j \frac{\ell^\star_{ij}}{\ell_{ij}}(\phi_j - \phi_i)$$



Cotangent Weights



$$\frac{\ell^\star}{\ell} = \frac{1}{2}(\cot\alpha + \cot\beta)$$

Discrete Poisson Equation

Continuous: $\Delta\phi = \rho$

Discrete: $\frac{1}{2}\sum_j (\cot\alpha_{ij} + \cot\beta_{ij})(\phi_j - \phi_i) = \mathcal{V}_i\rho_i$

Matrix: $L\phi = M\rho$

Properties of the (Discrete) Laplacian

Properties of the (Discrete) Laplacian

- Sparse

Properties of the (Discrete) Laplacian

- Sparse
- Symmetric

Properties of the (Discrete) Laplacian

- Sparse
- Symmetric
- Positive-semidefinite ($x^{\mathrm{T}}Lx \geq 0$)

Properties of the (Discrete) Laplacian

- Sparse
- Symmetric
- Positive-semidefinite ($x^{\mathsf{T}} L x \geq 0$)
- Constants in kernel / cokernel

Properties of the (Discrete) Laplacian

- Sparse
- Symmetric
- Positive-semidefinite ($x^{\mathsf{T}}Lx \geq 0$)
- Constants in kernel/cokernel
- Weak vs. strong

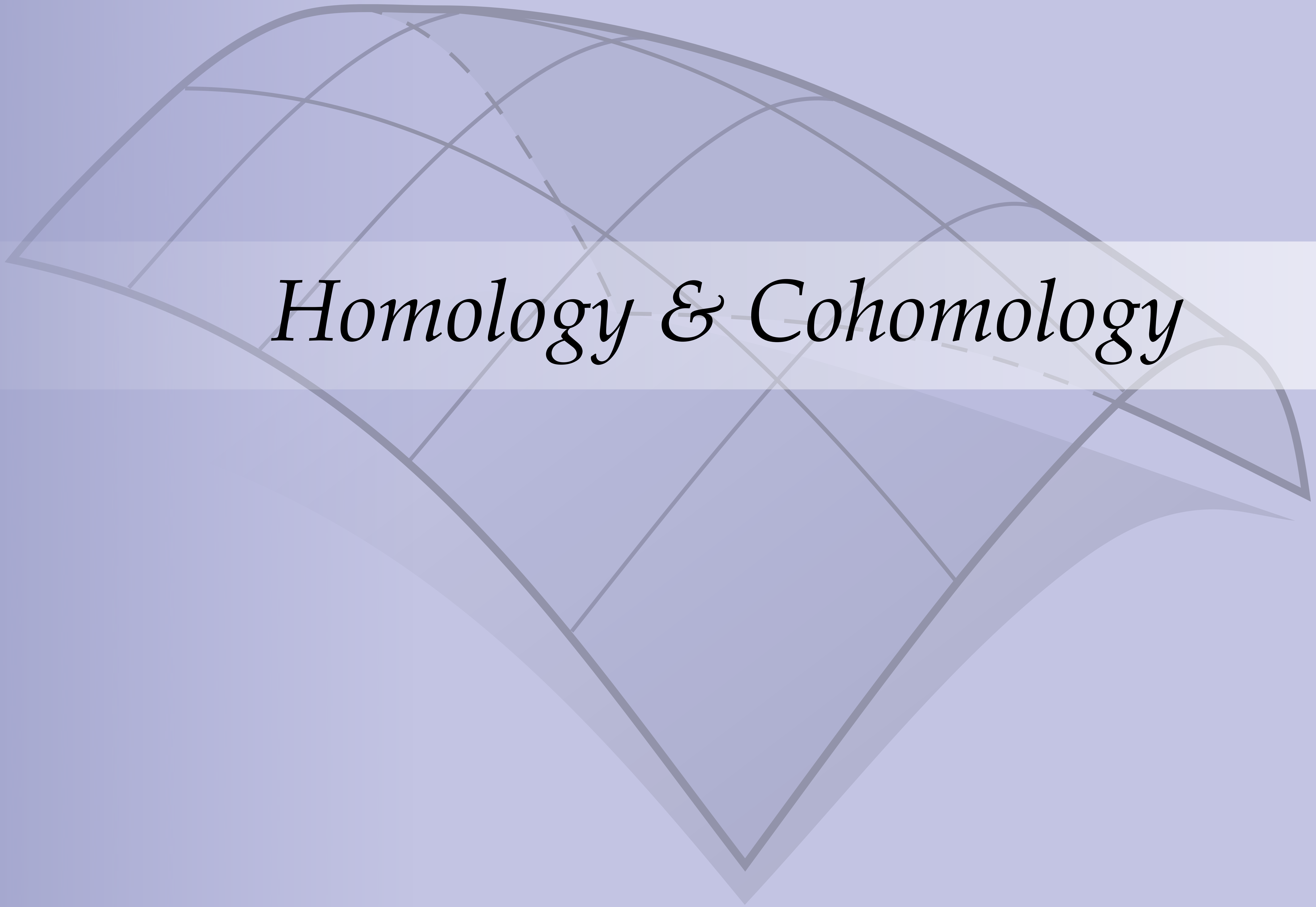
Properties of the (Discrete) Laplacian

- Sparse
- Symmetric
- Positive-semidefinite ($x^{\mathsf{T}}Lx \geq 0$)
- Constants in kernel/cokernel
- Weak vs. strong
- maximum principle? (elliptic vs. hyperbolic)

Properties of the (Discrete) Laplacian

- Sparse
- Symmetric
- Positive-semidefinite ($x^{\mathsf{T}}Lx \geq 0$)
- Constants in kernel/cokernel
- Weak vs. strong
- maximum principle? (elliptic vs. hyperbolic)
- *Extremely useful in geometry processing!*

Homology & Cohomology



Homology & Cohomology

Homology & Cohomology

$$\partial \circ \partial = \emptyset$$

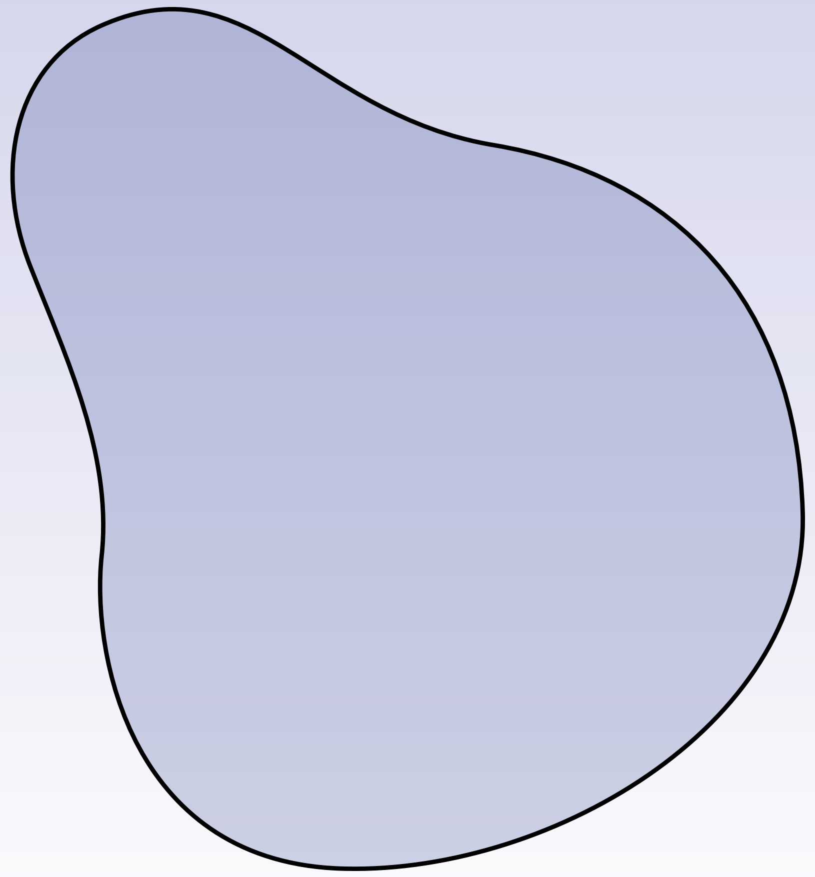
Homology & Cohomology

$$\partial \circ \partial = \emptyset$$

$$d \circ d = 0$$

Homology & Cohomology

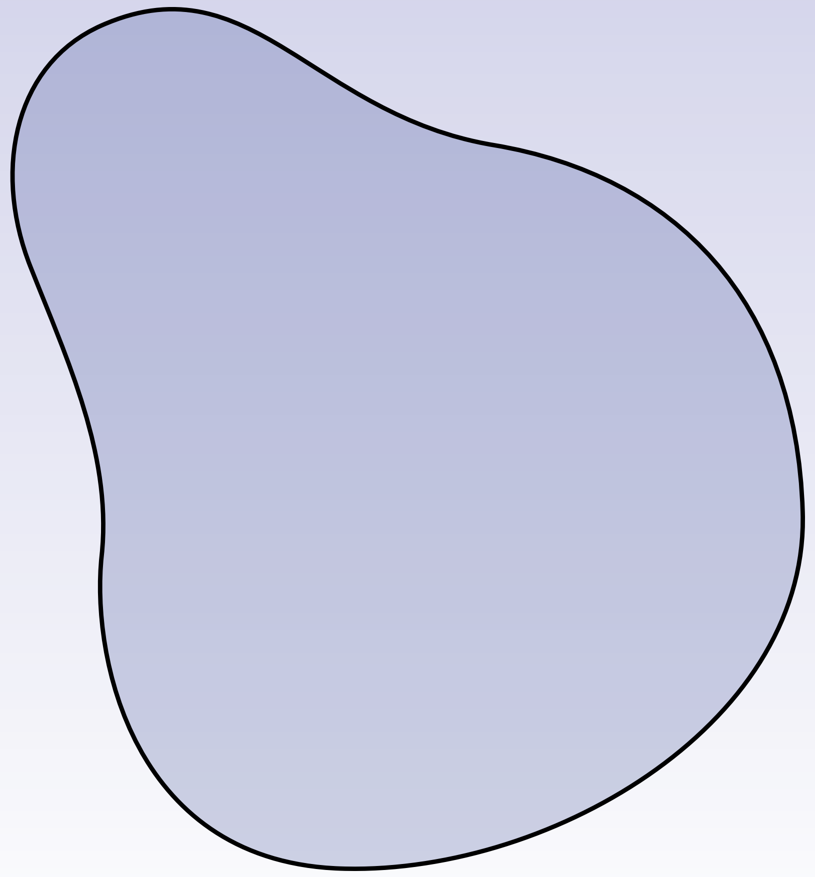
$$\partial \circ \partial = \emptyset$$



$$d \circ d = 0$$

Homology & Cohomology

$$\partial \circ \partial = \emptyset$$

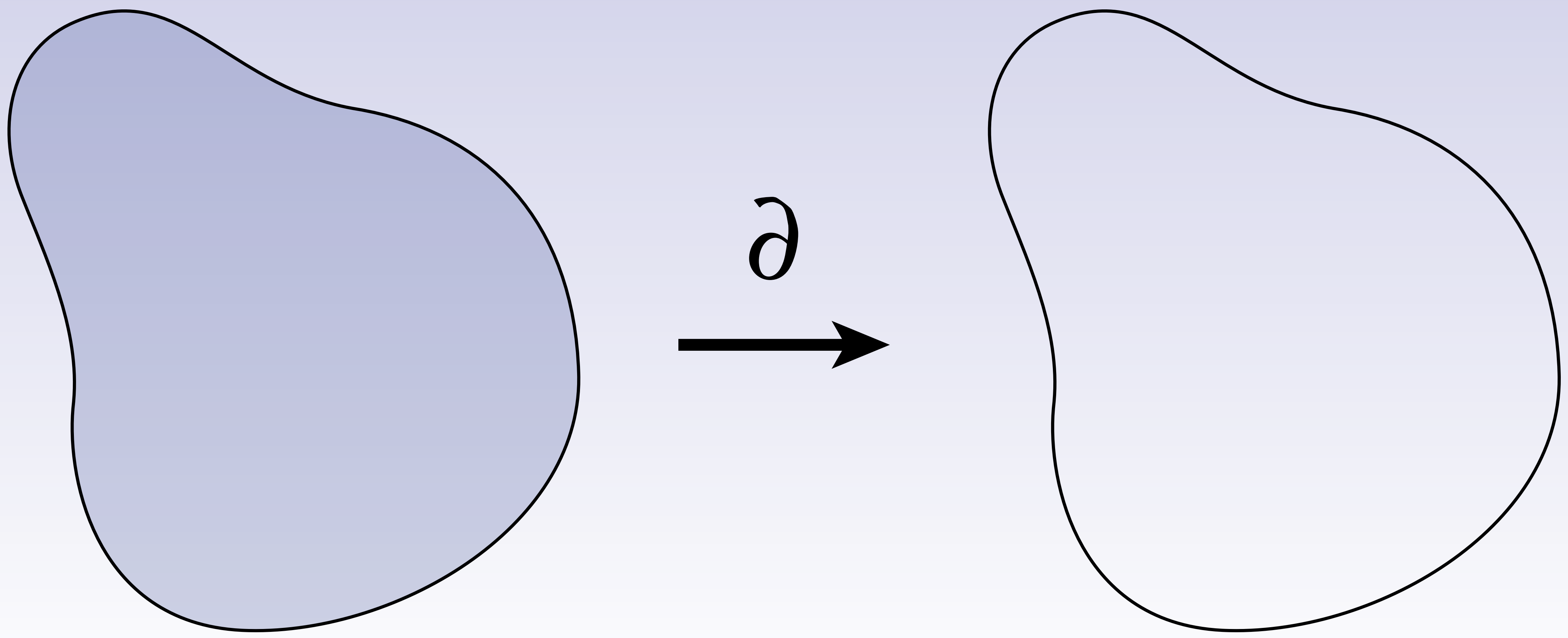


$$\xrightarrow{\partial}$$

$$d \circ d = 0$$

Homology & Cohomology

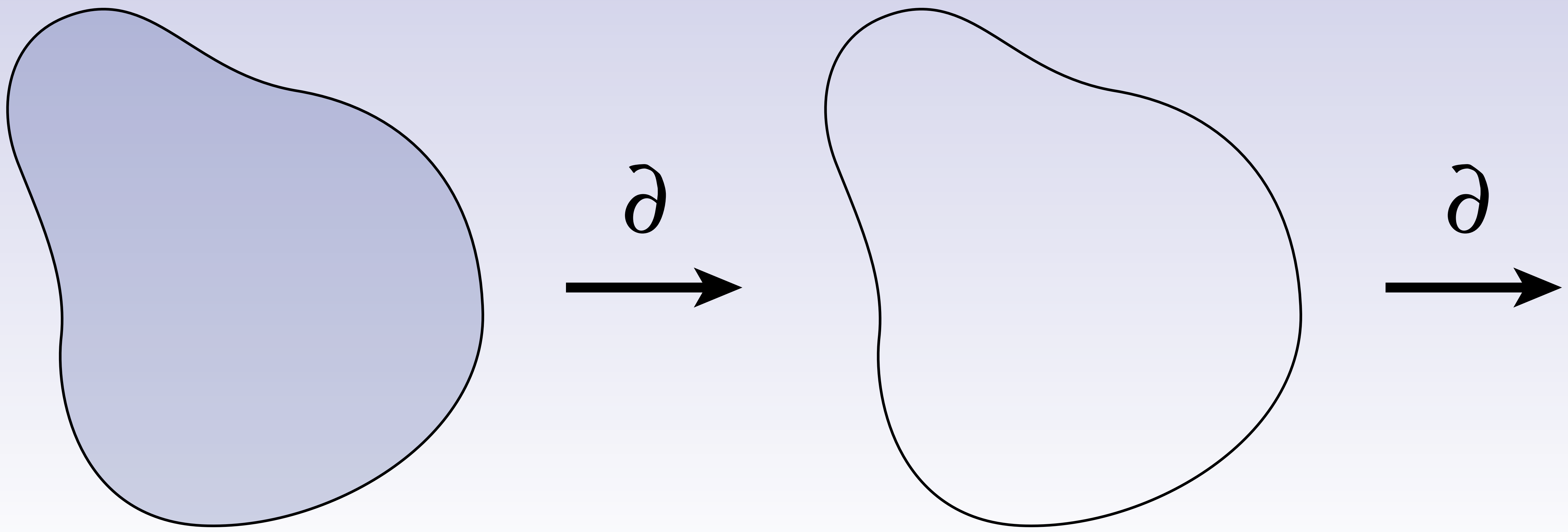
$$\partial \circ \partial = \emptyset$$



$$d \circ d = 0$$

Homology & Cohomology

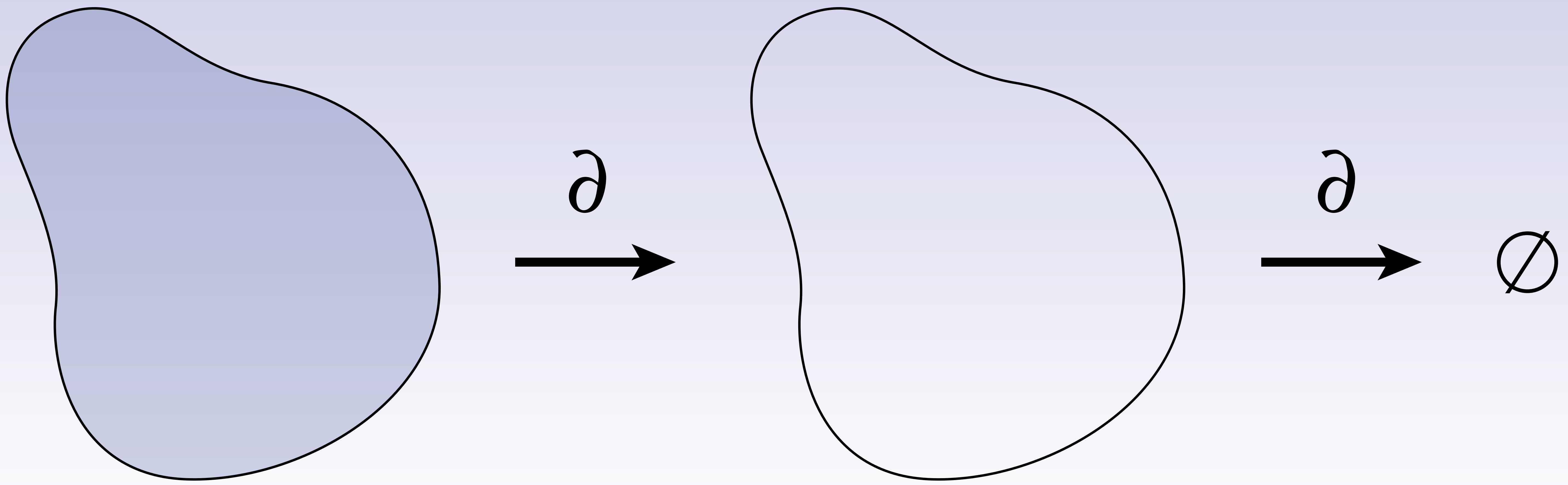
$$\partial \circ \partial = \emptyset$$



$$d \circ d = 0$$

Homology & Cohomology

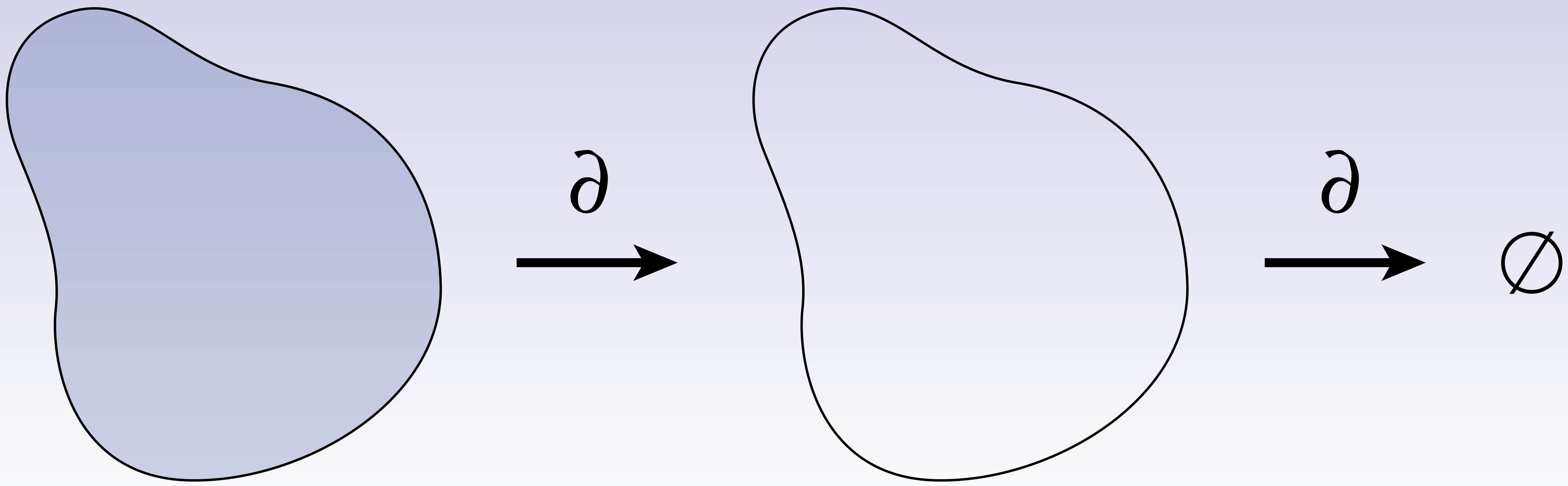
$$\partial \circ \partial = \emptyset$$



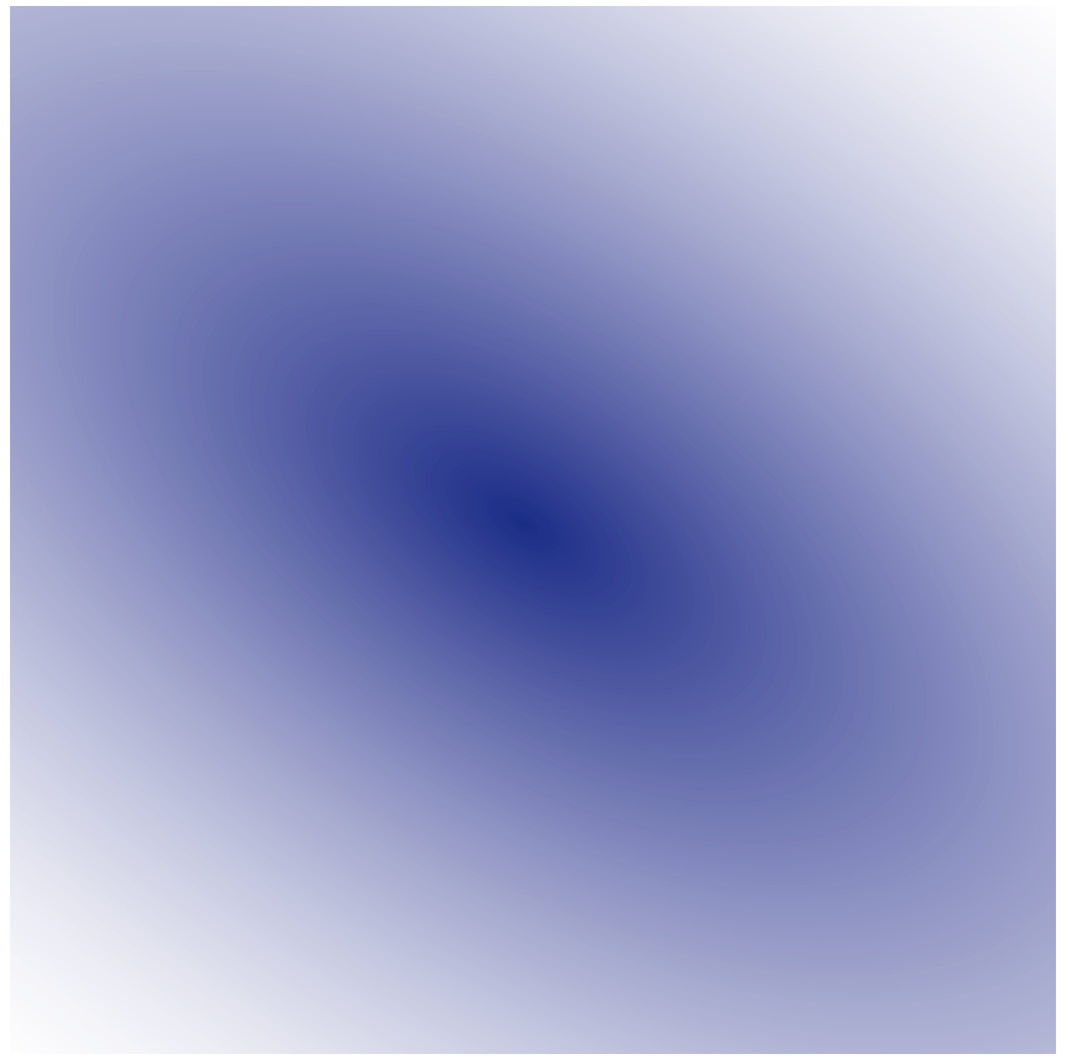
$$d \circ d = 0$$

Homology & Cohomology

$\partial \circ \partial = \emptyset$



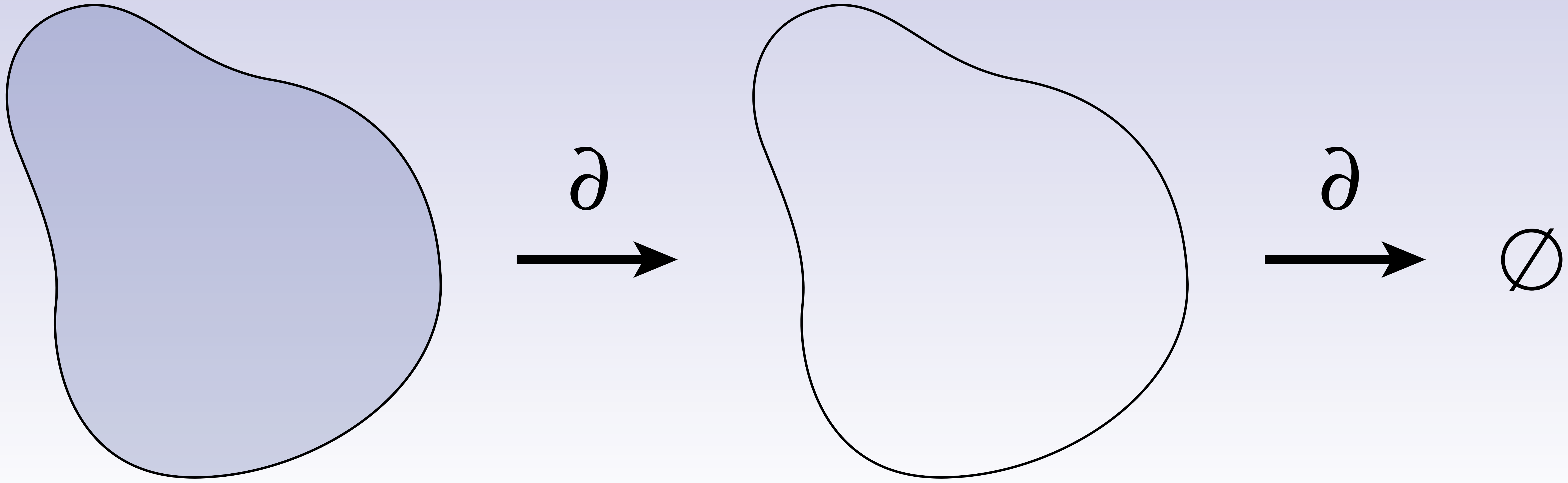
$d \circ d = 0$



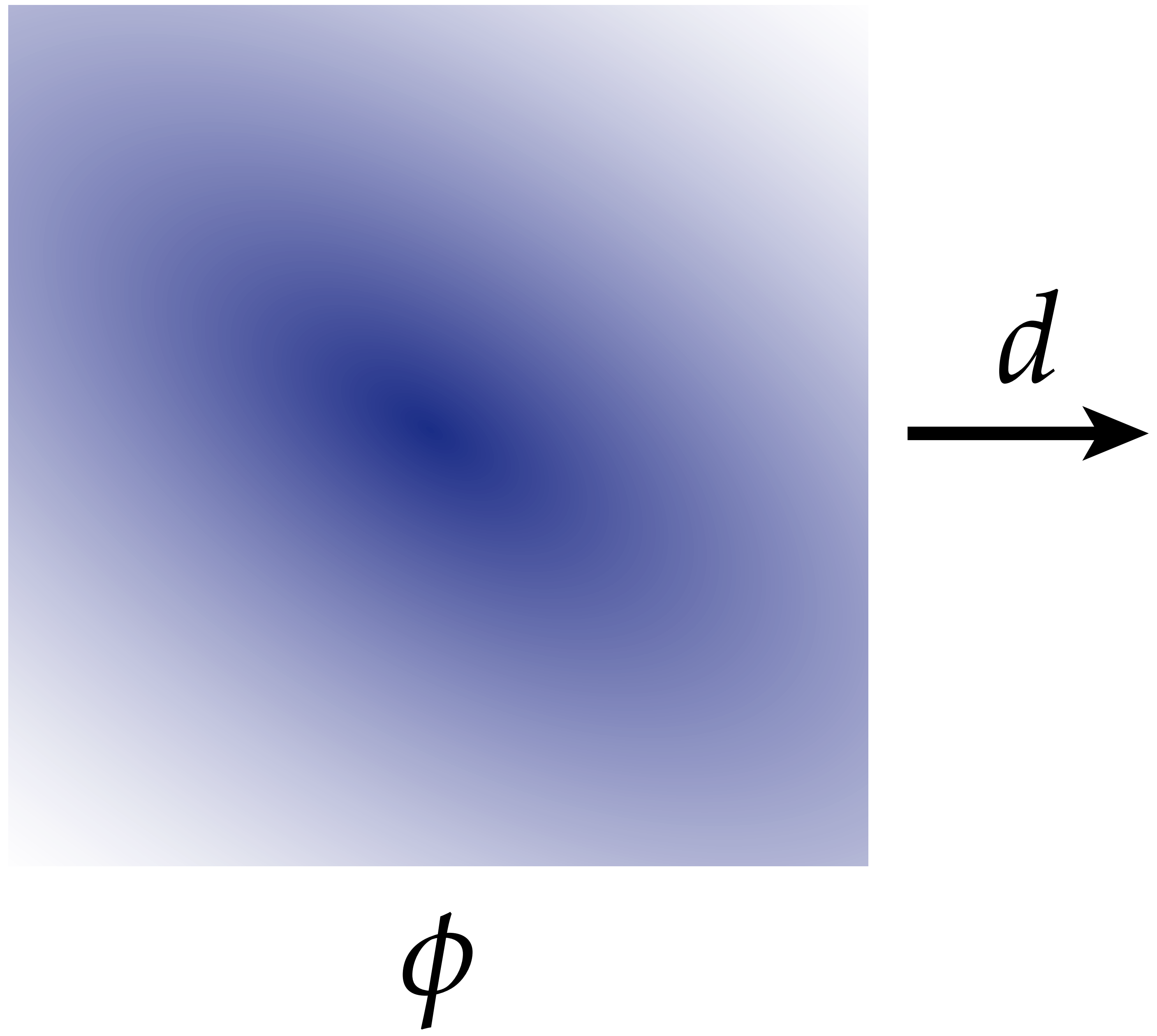
ϕ

Homology & Cohomology

$\partial \circ \partial = \varnothing$

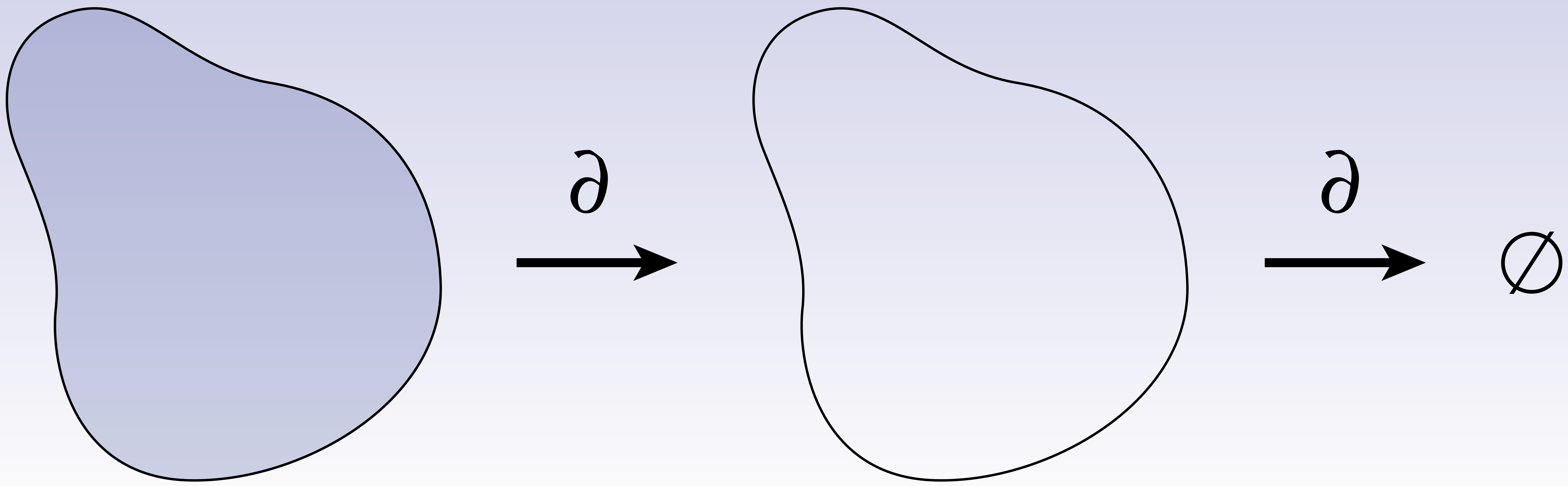


$d \circ d = 0$

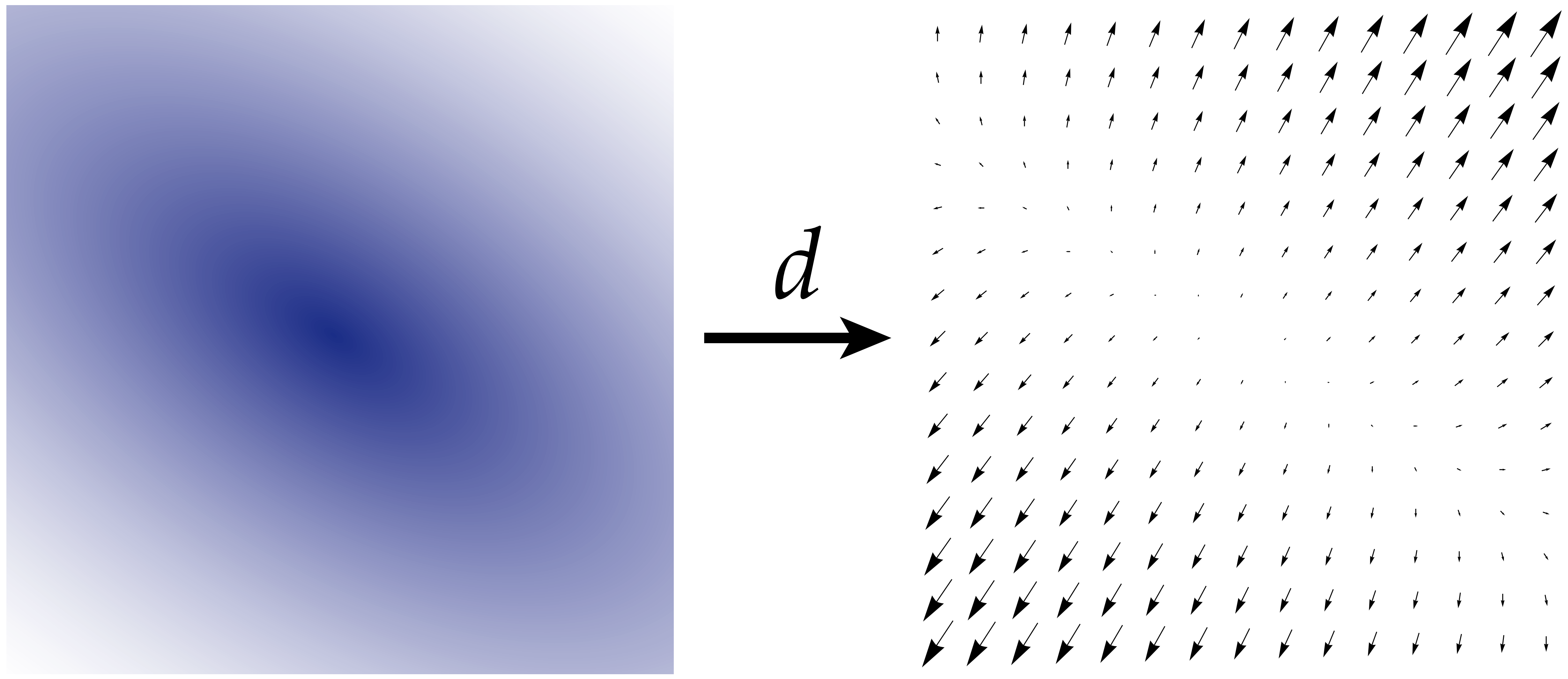


Homology & Cohomology

$$\partial \circ \partial = \emptyset$$



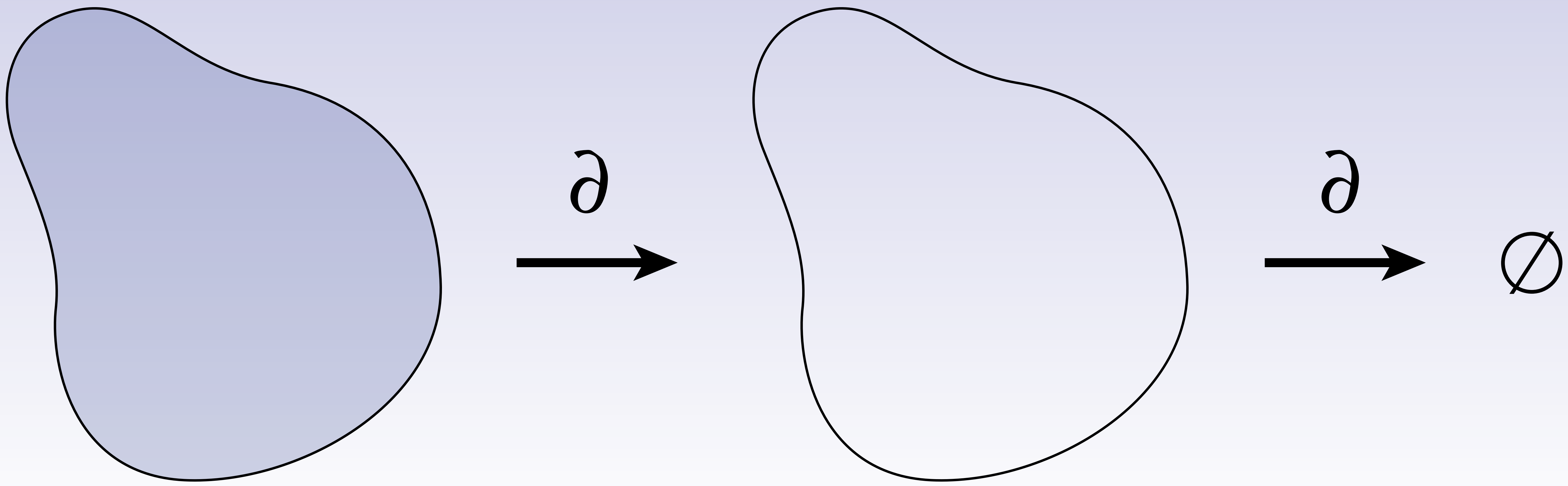
$$d \circ d = 0$$



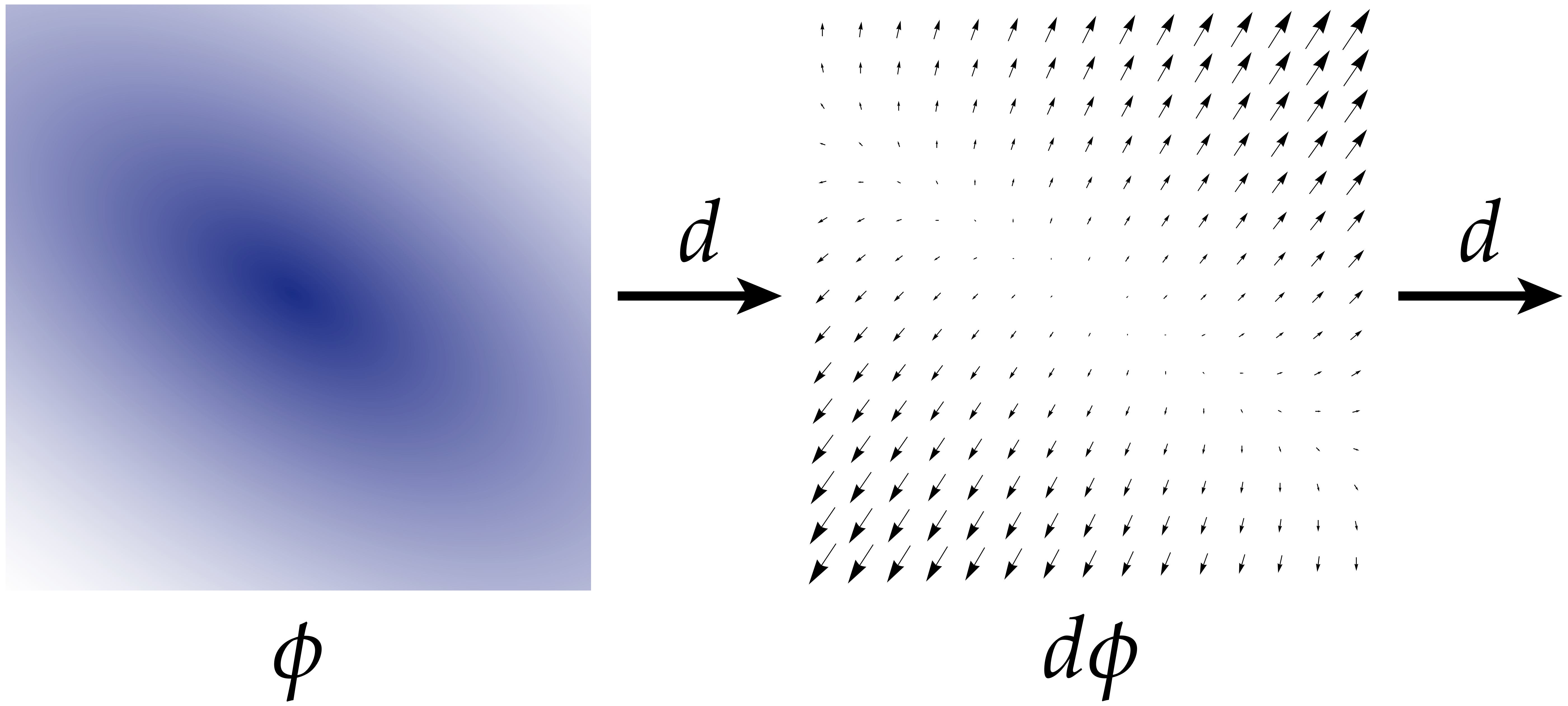
ϕ $\quad$ $d\phi$

Homology & Cohomology

$\partial \circ \partial = \emptyset$

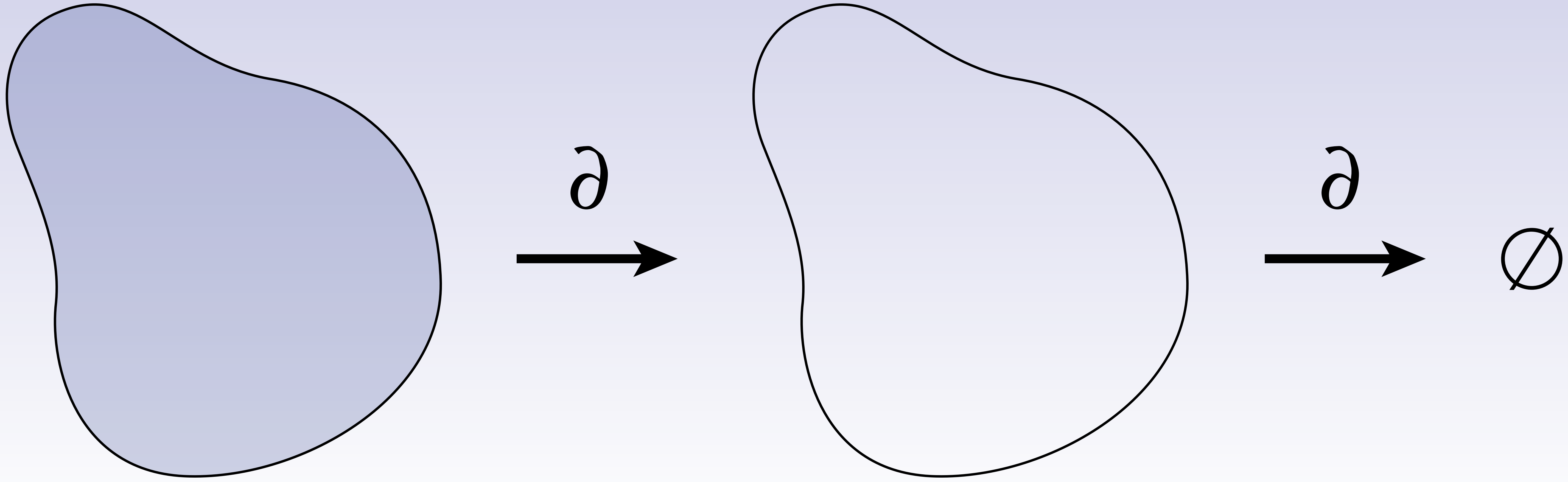


$d \circ d = 0$

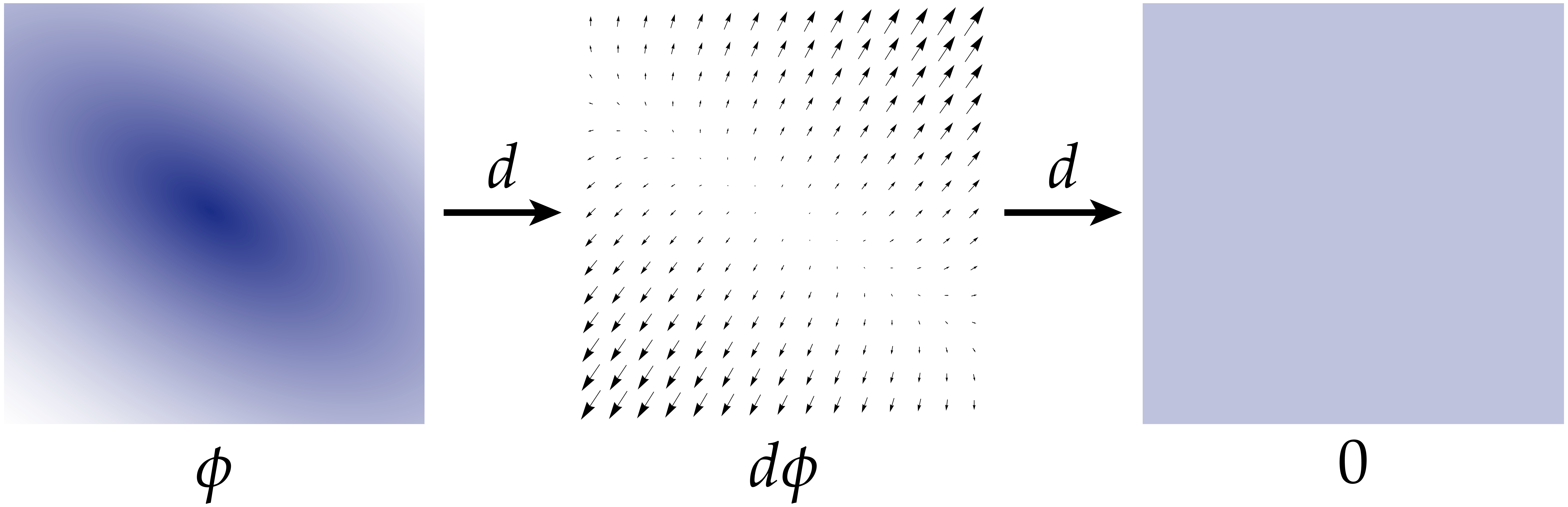


Homology & Cohomology

$$\partial \circ \partial = \emptyset$$

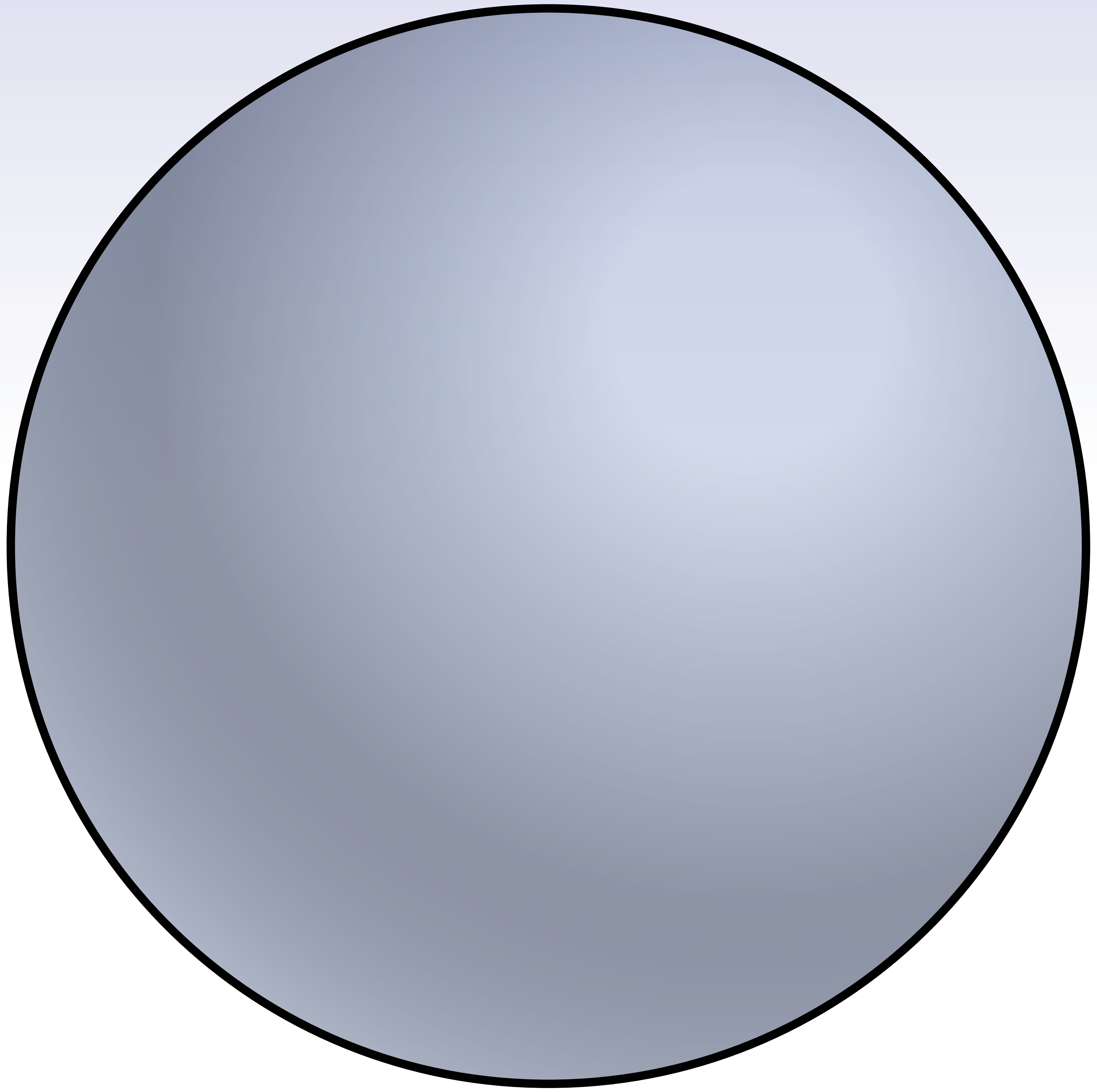


$$d \circ d = 0$$

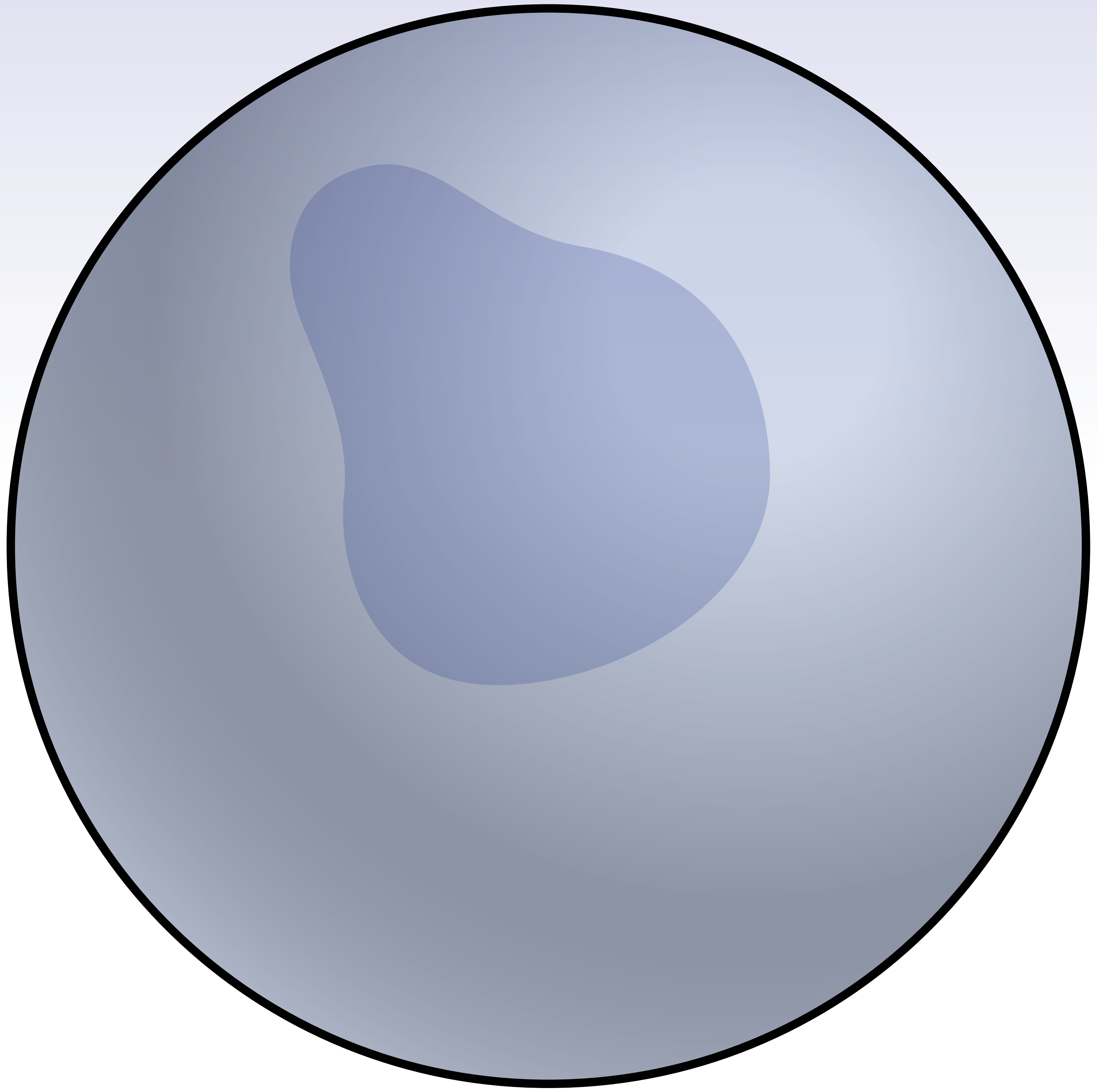


Homology on Surfaces

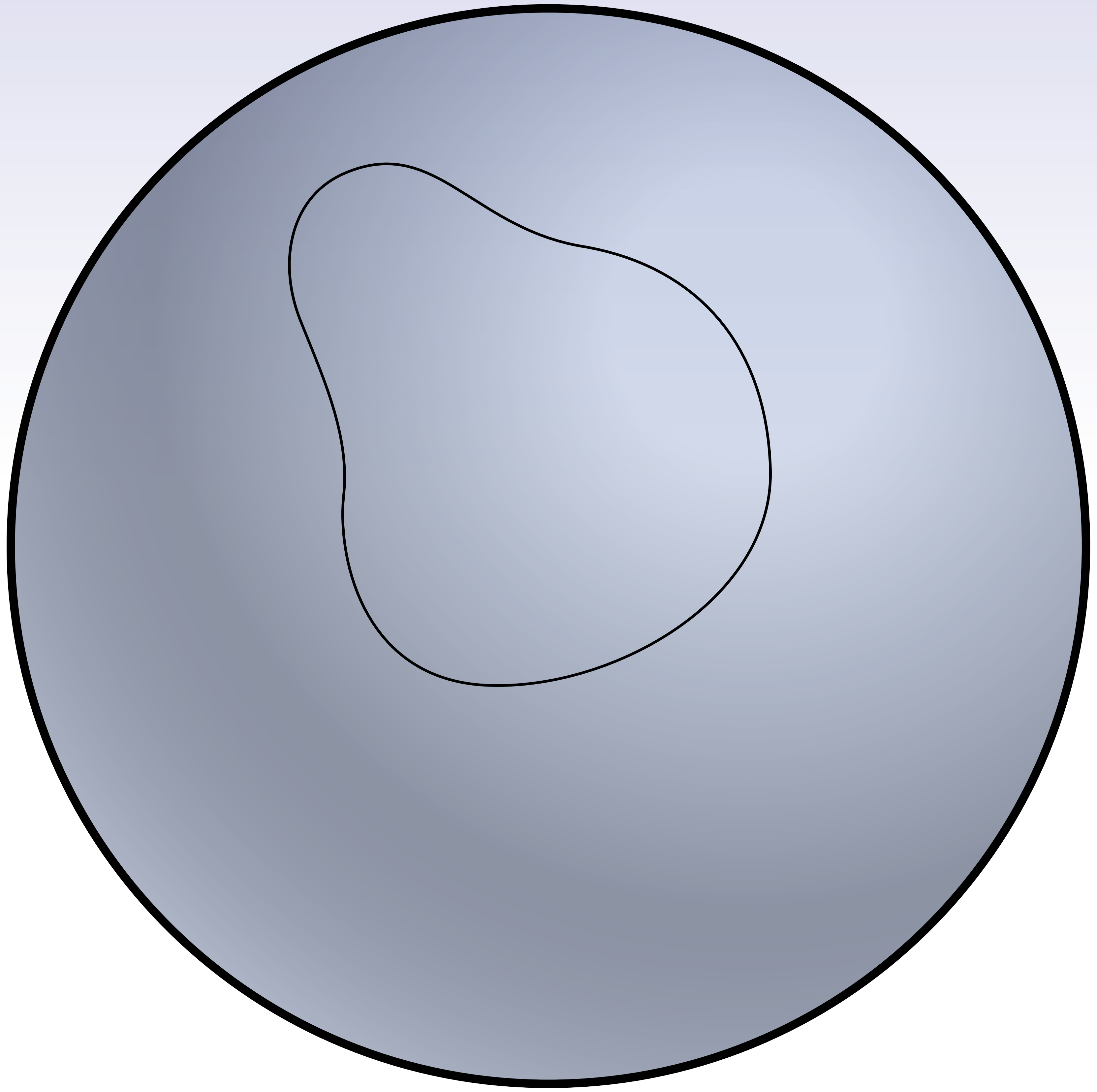
Homology on Surfaces



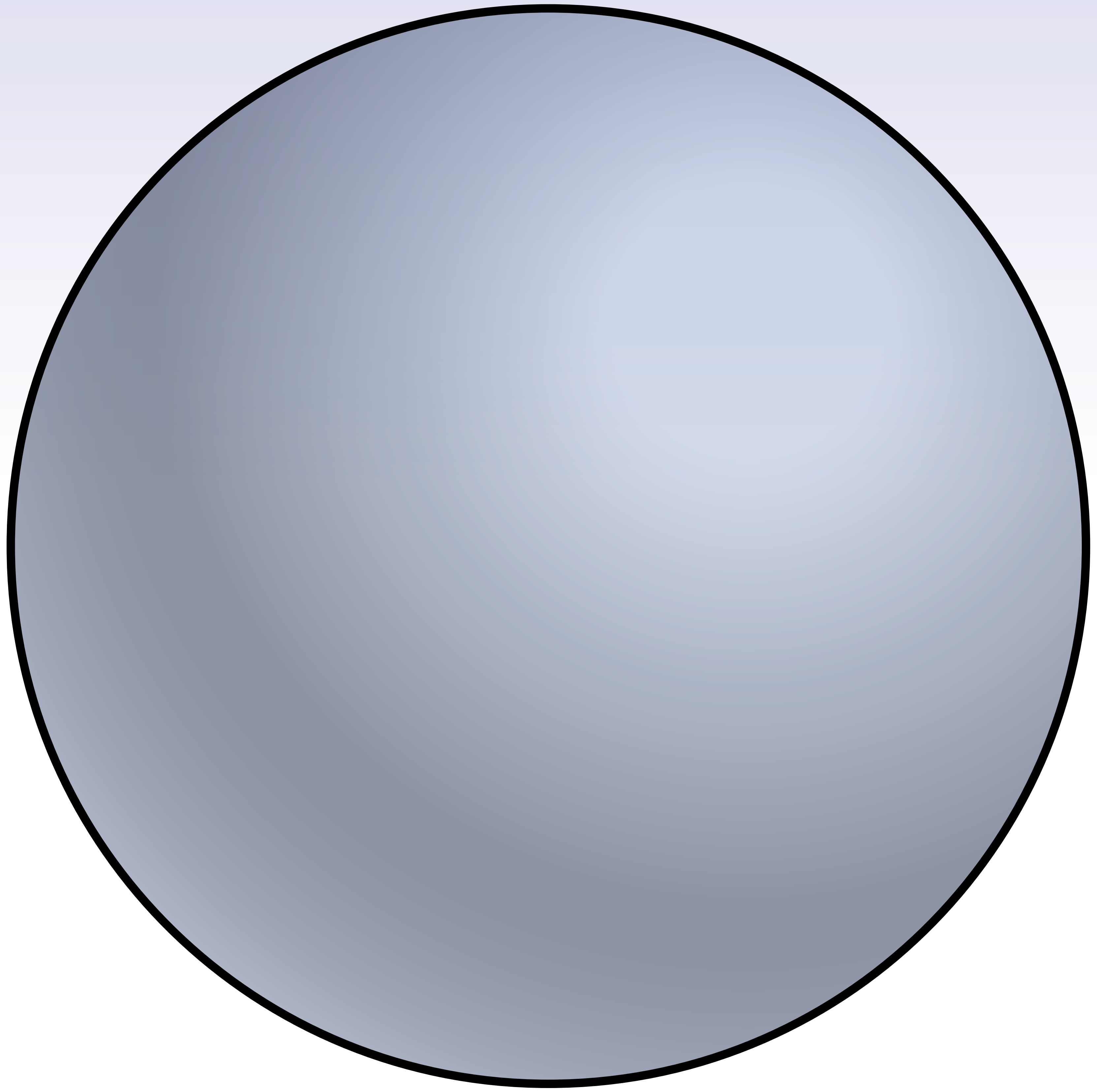
Homology on Surfaces



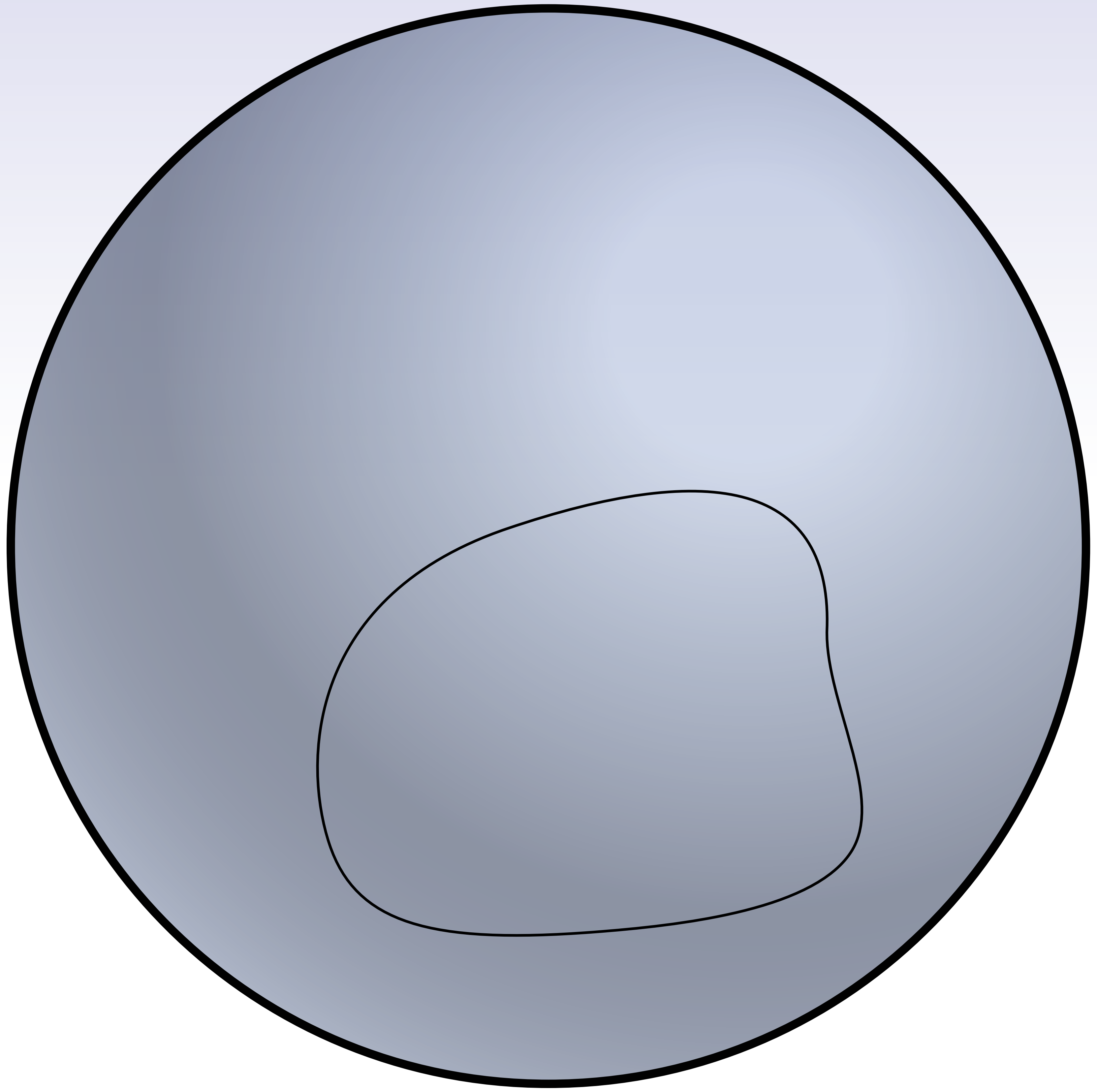
Homology on Surfaces



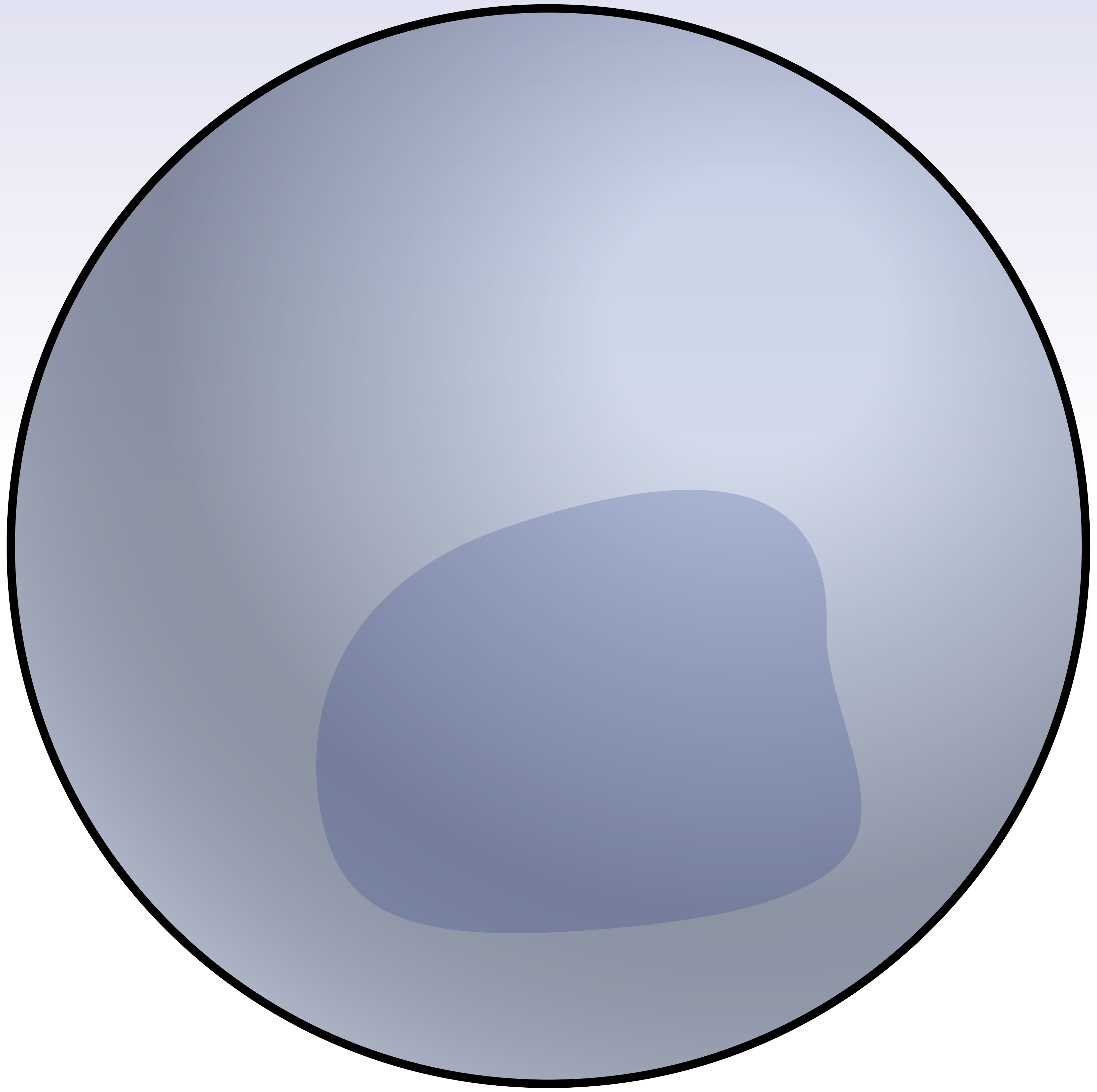
Homology on Surfaces



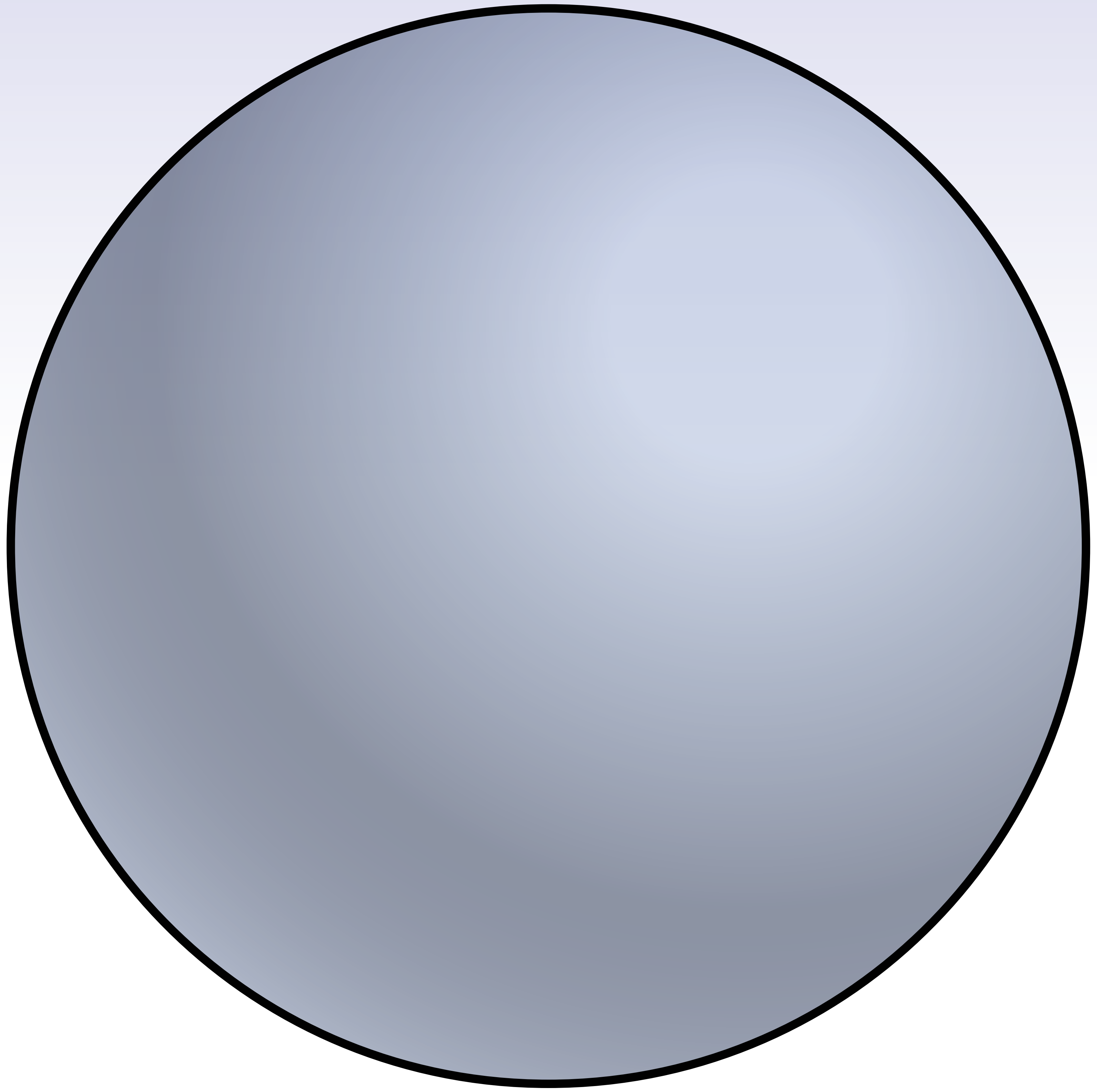
Homology on Surfaces



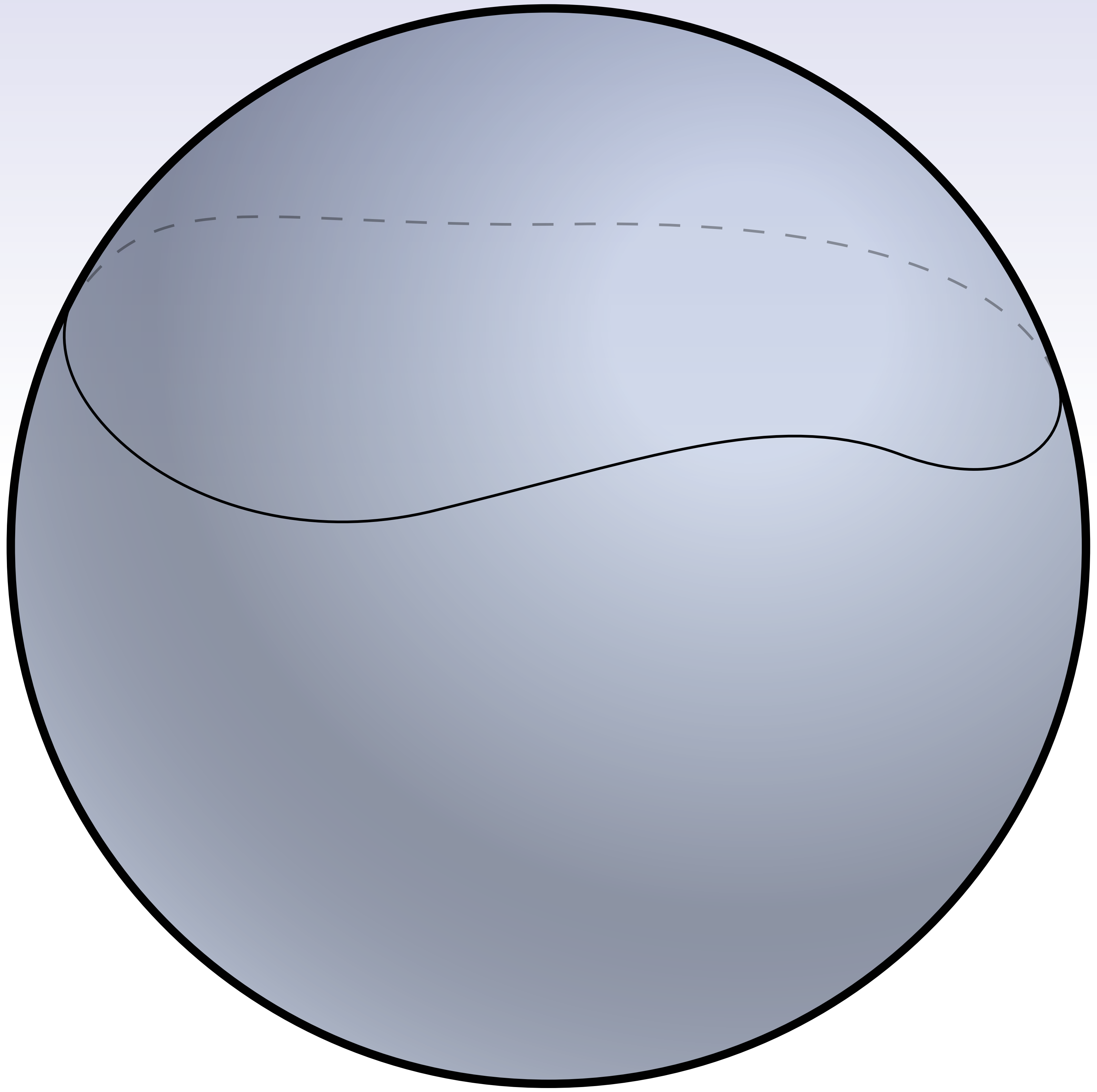
Homology on Surfaces



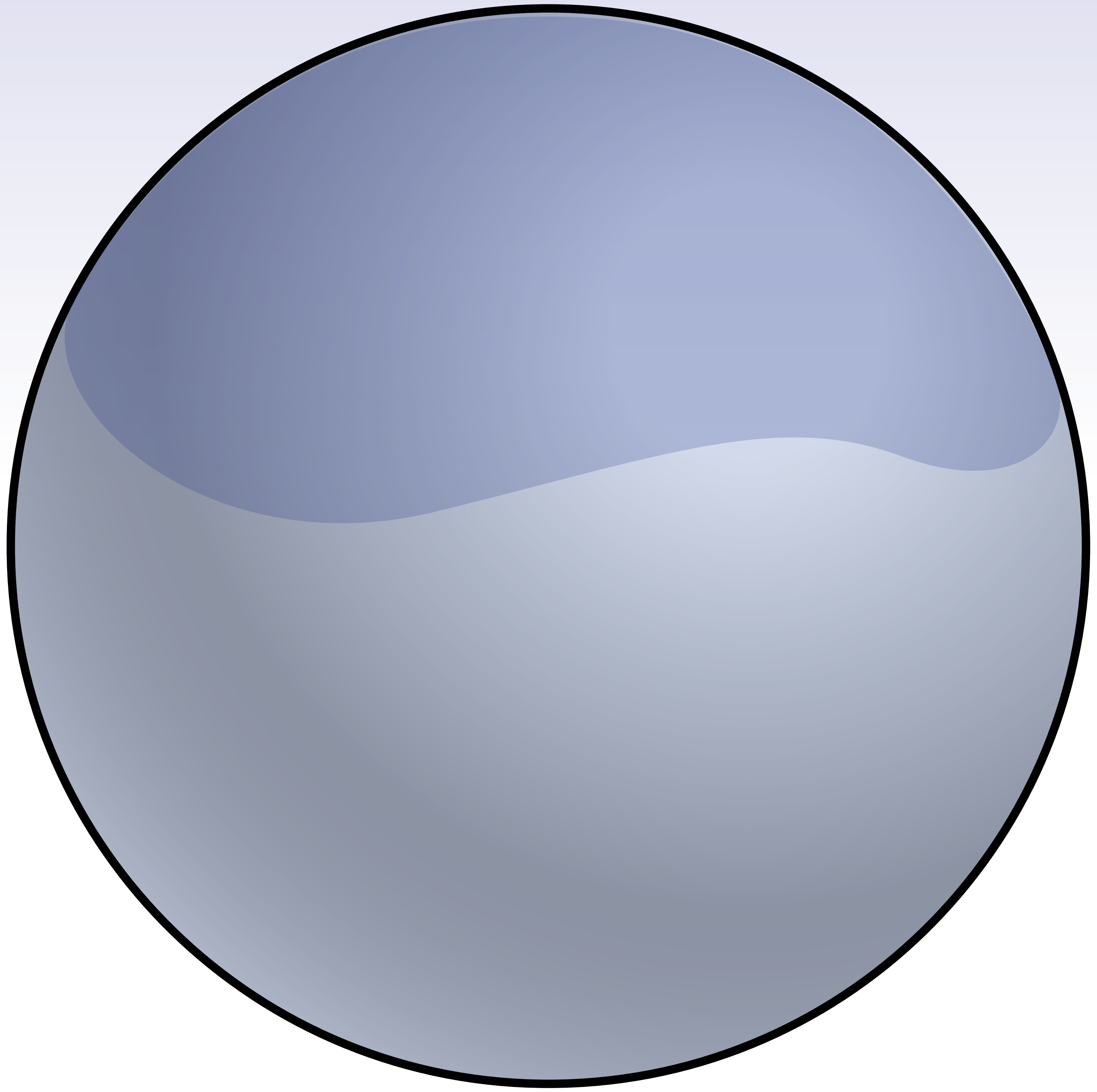
Homology on Surfaces



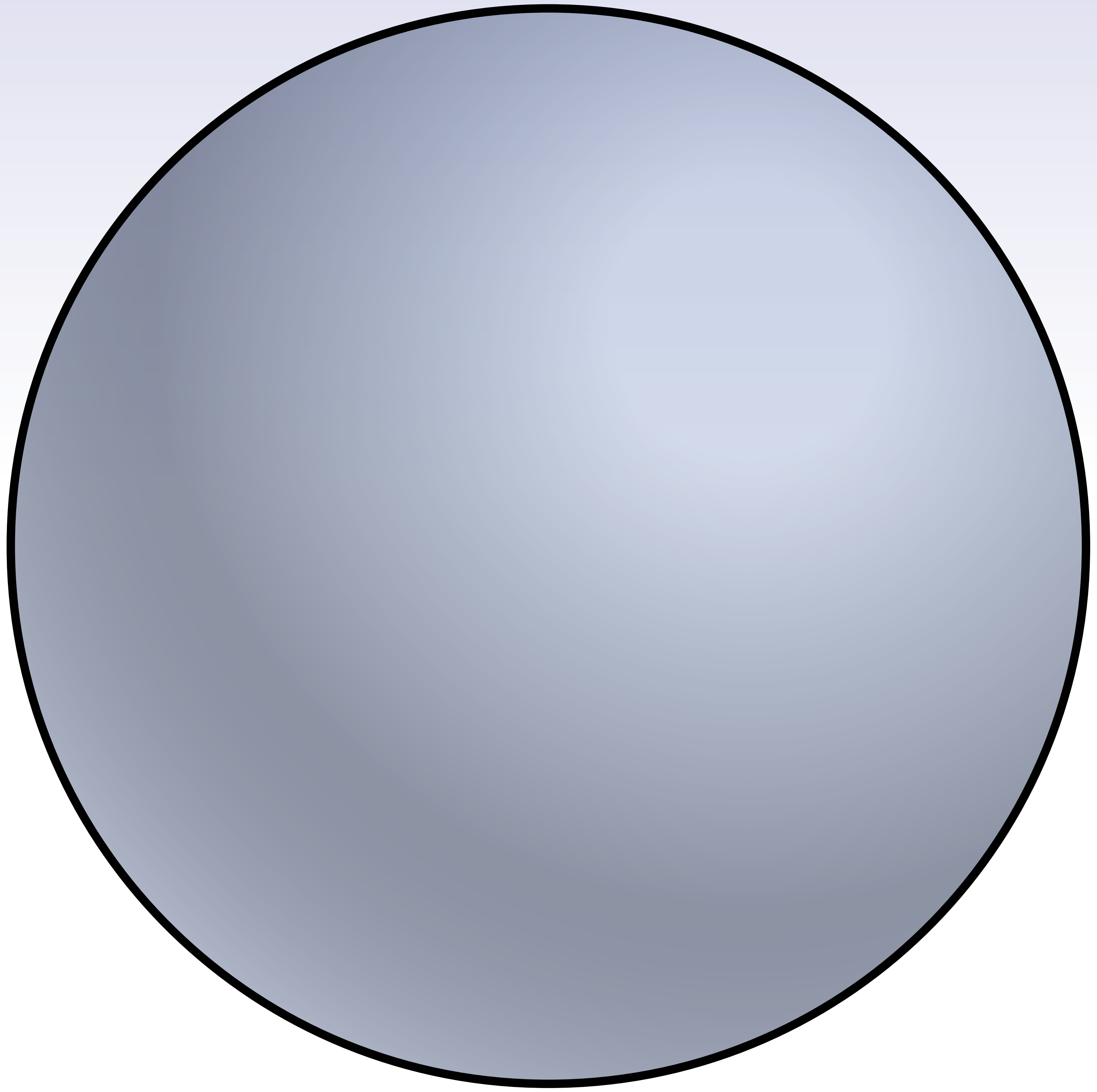
Homology on Surfaces



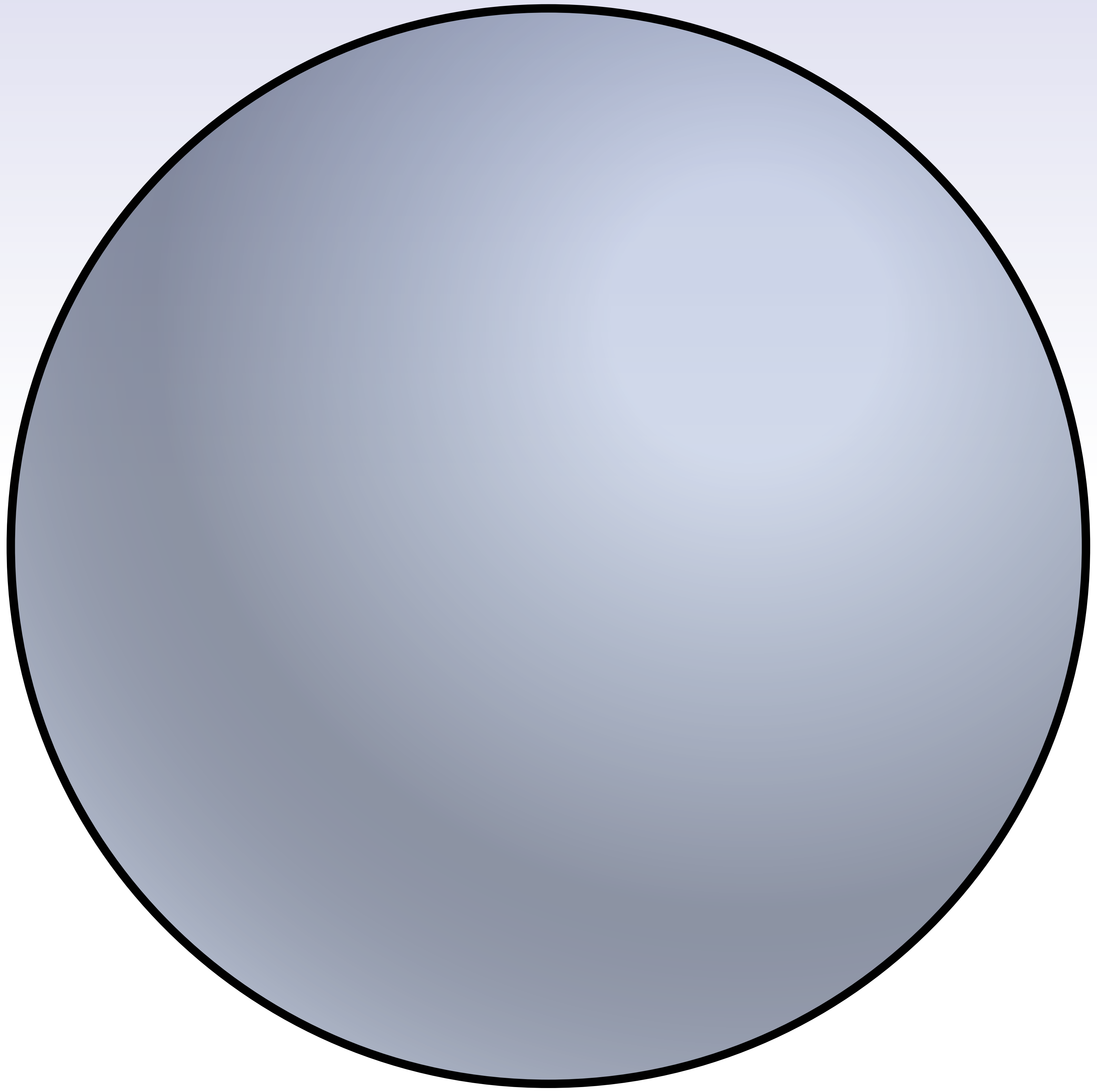
Homology on Surfaces

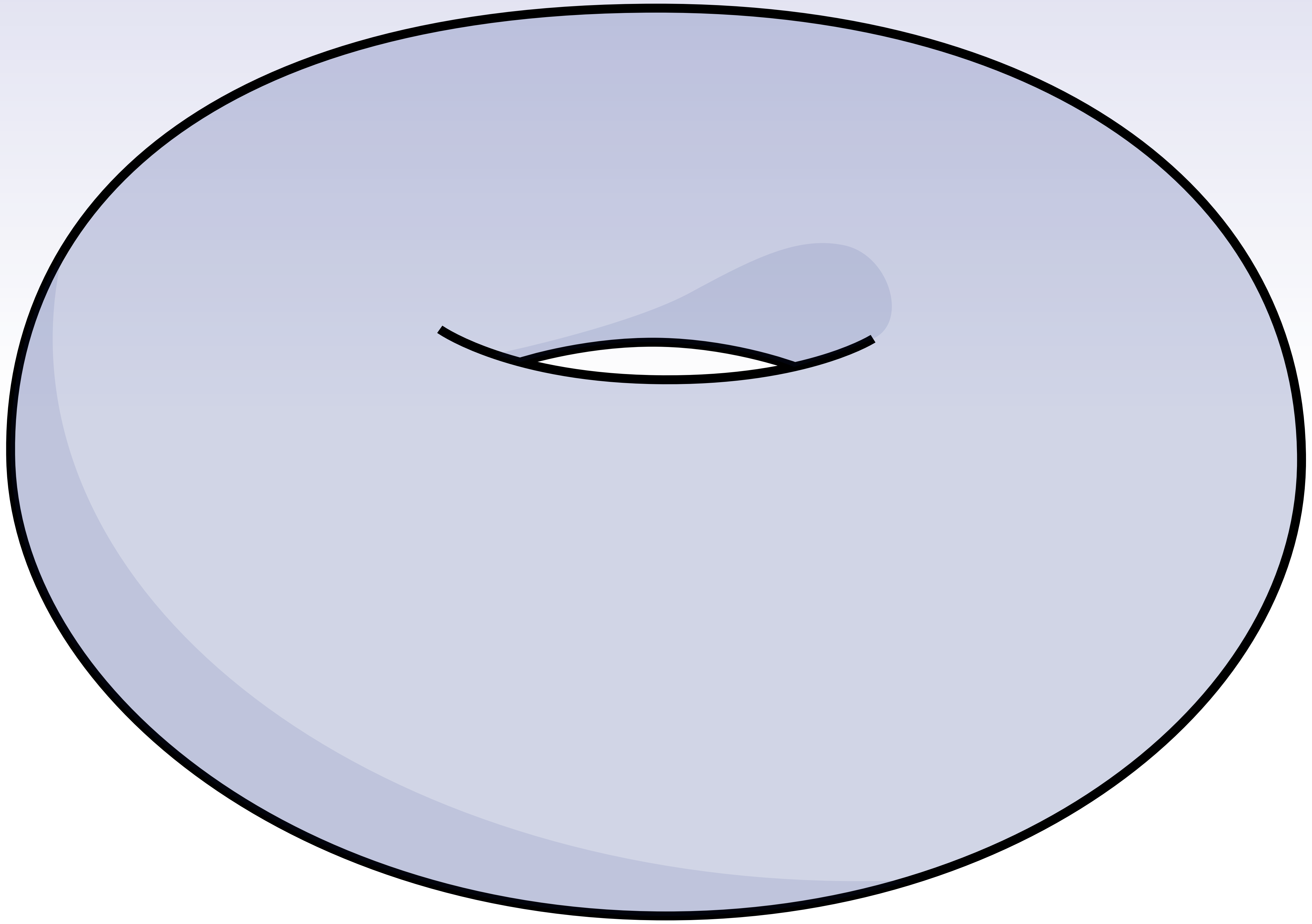


Homology on Surfaces

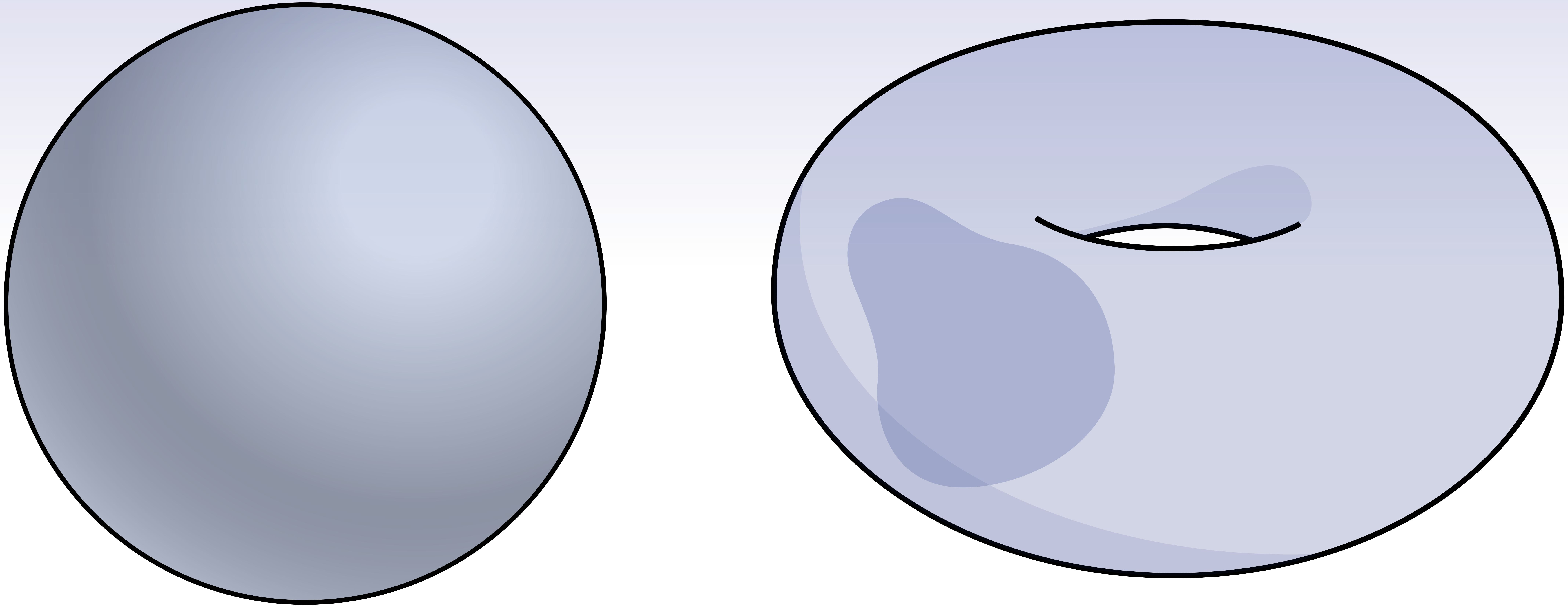


Homology on Surfaces

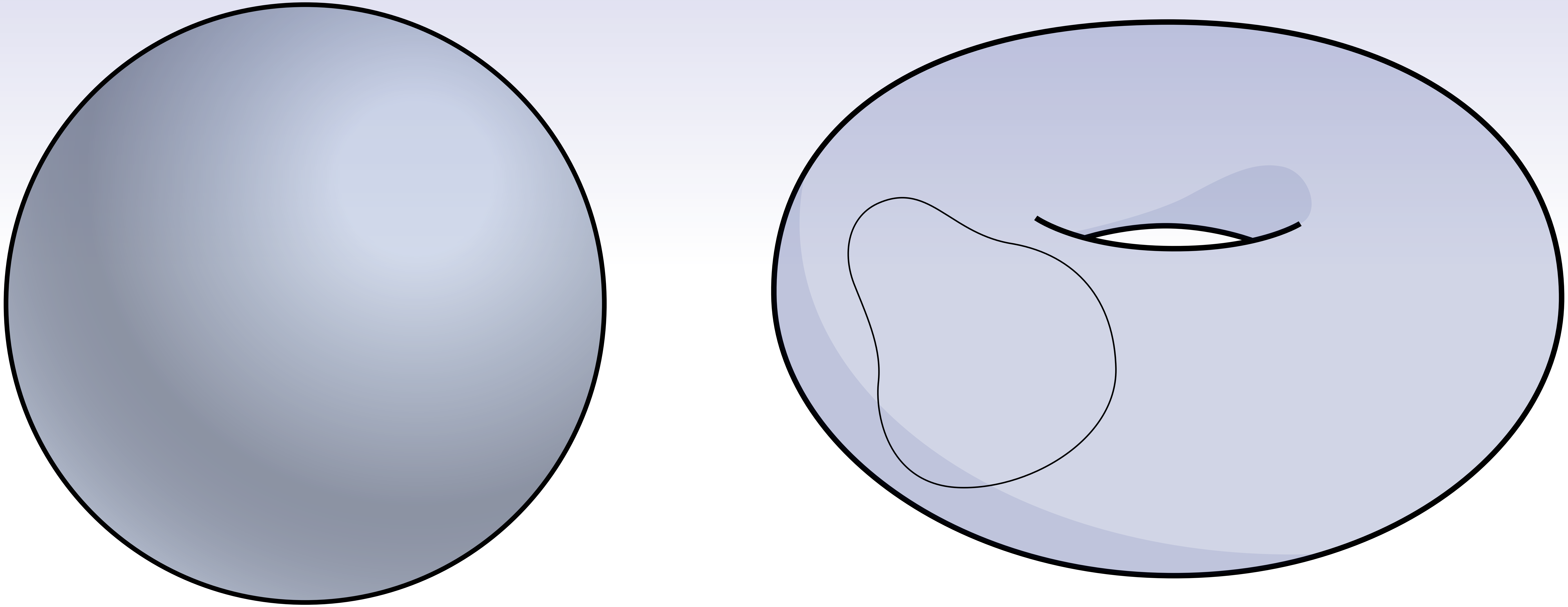




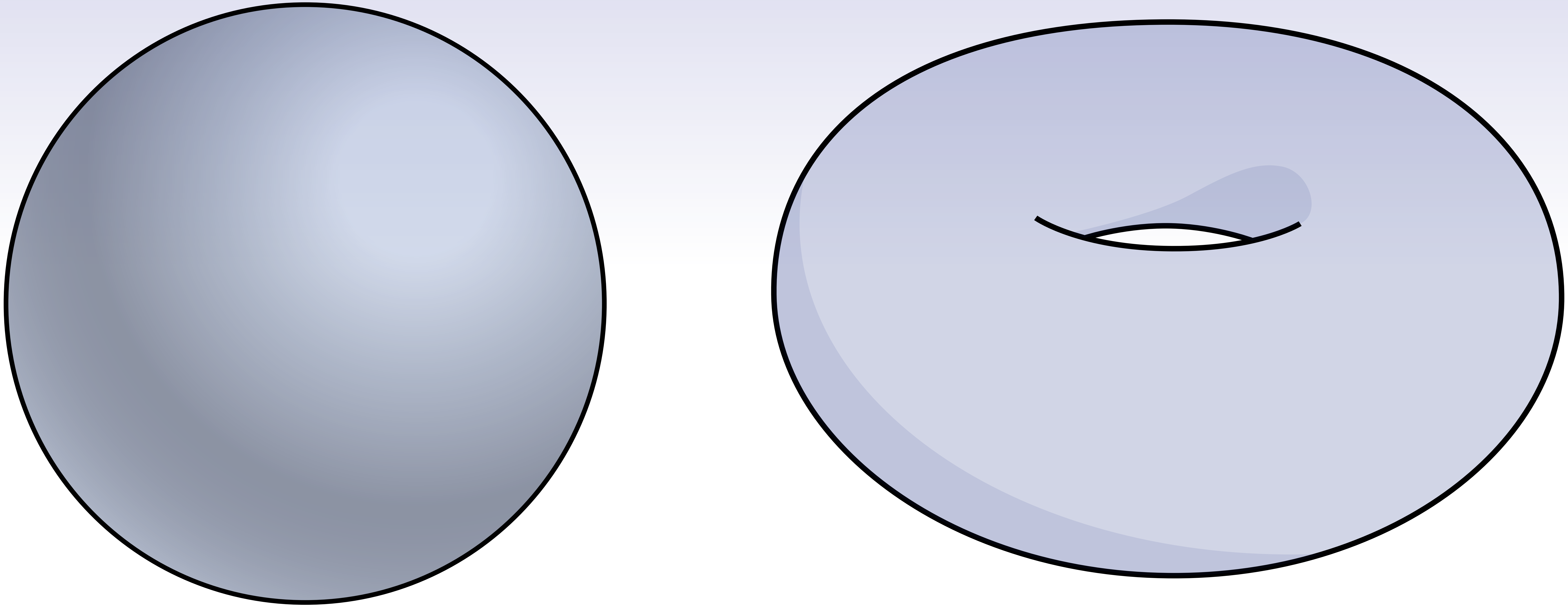
Homology on Surfaces



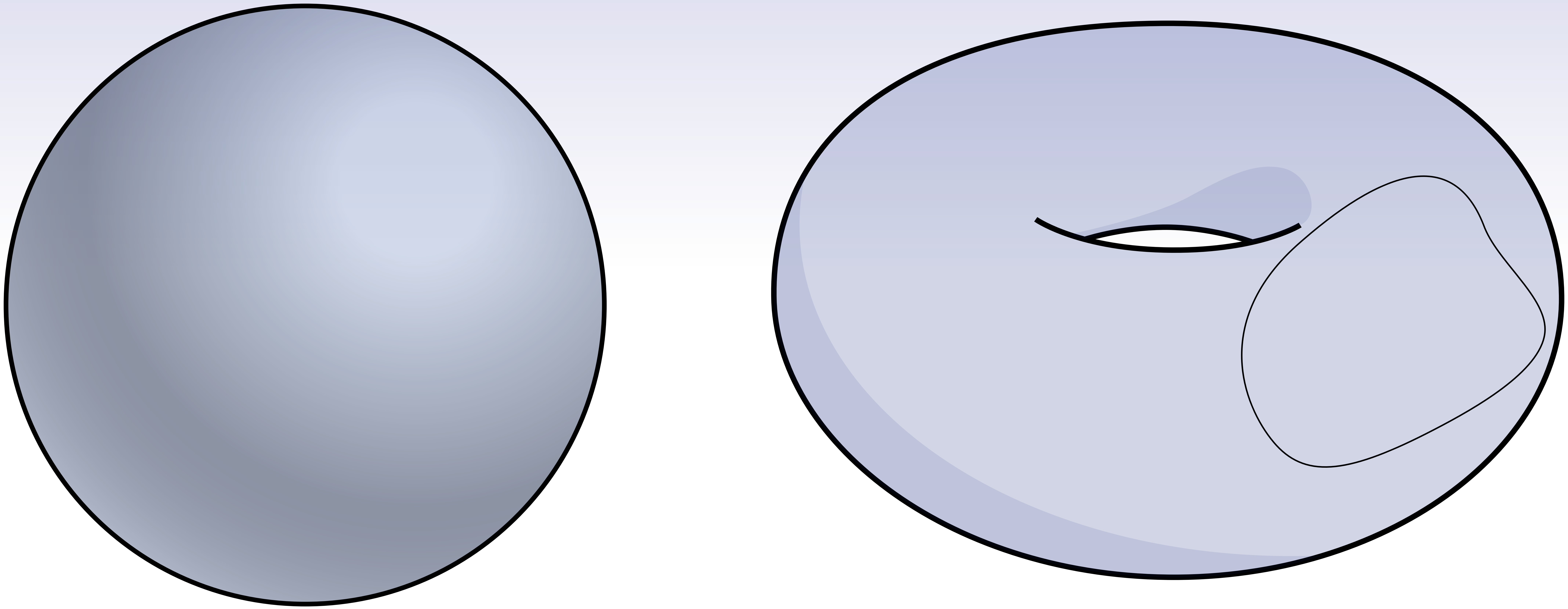
Homology on Surfaces



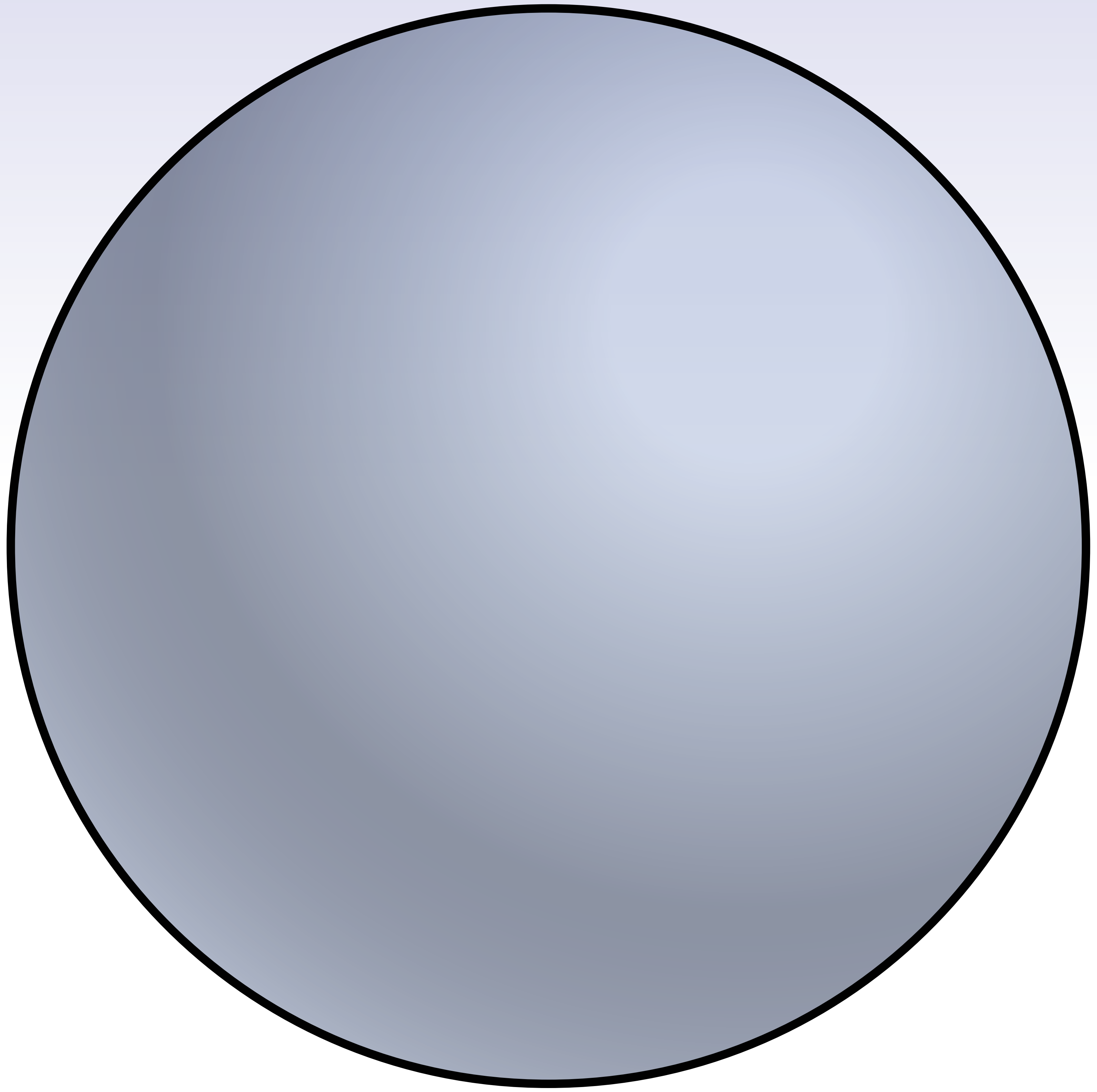
Homology on Surfaces

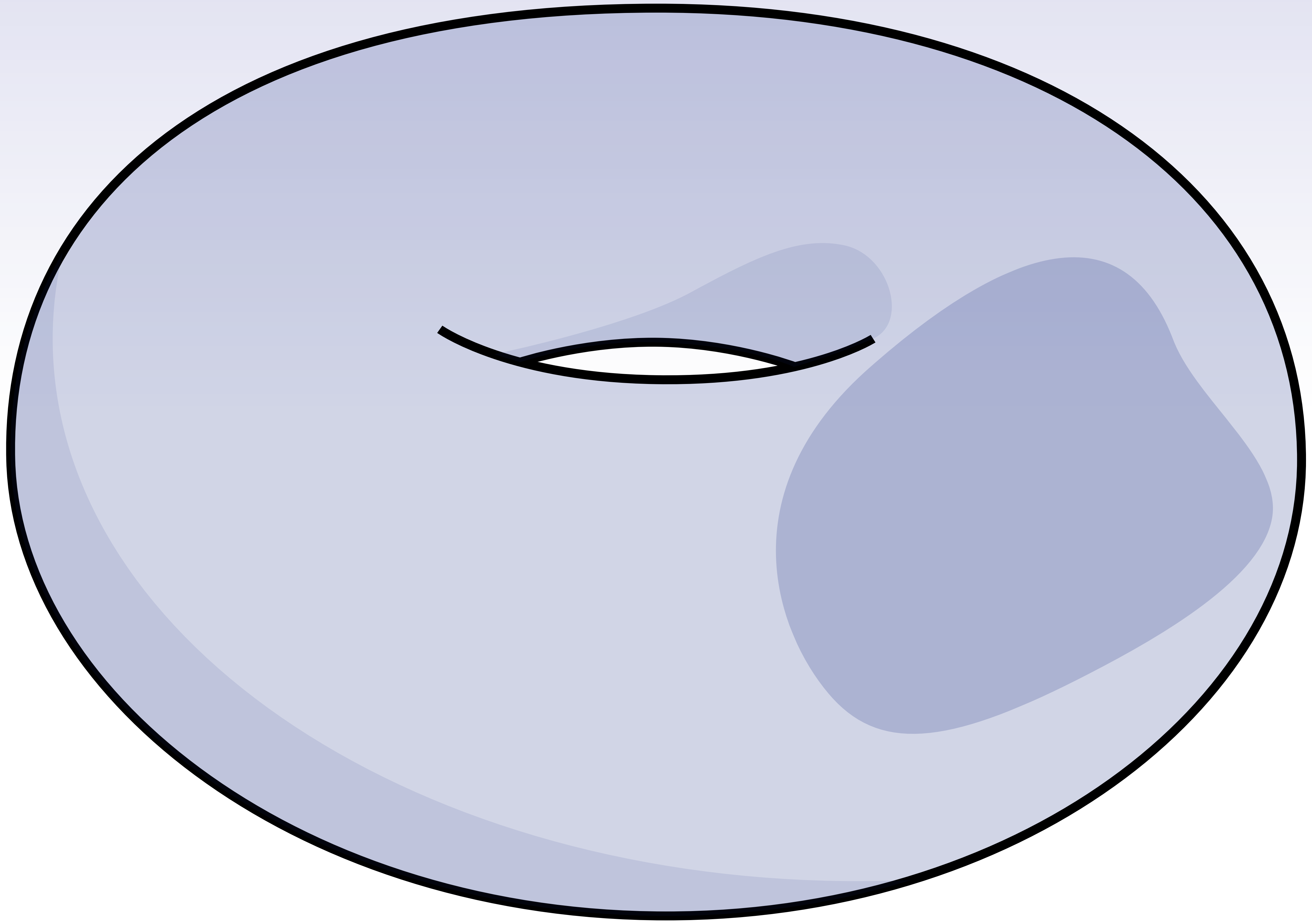


Homology on Surfaces

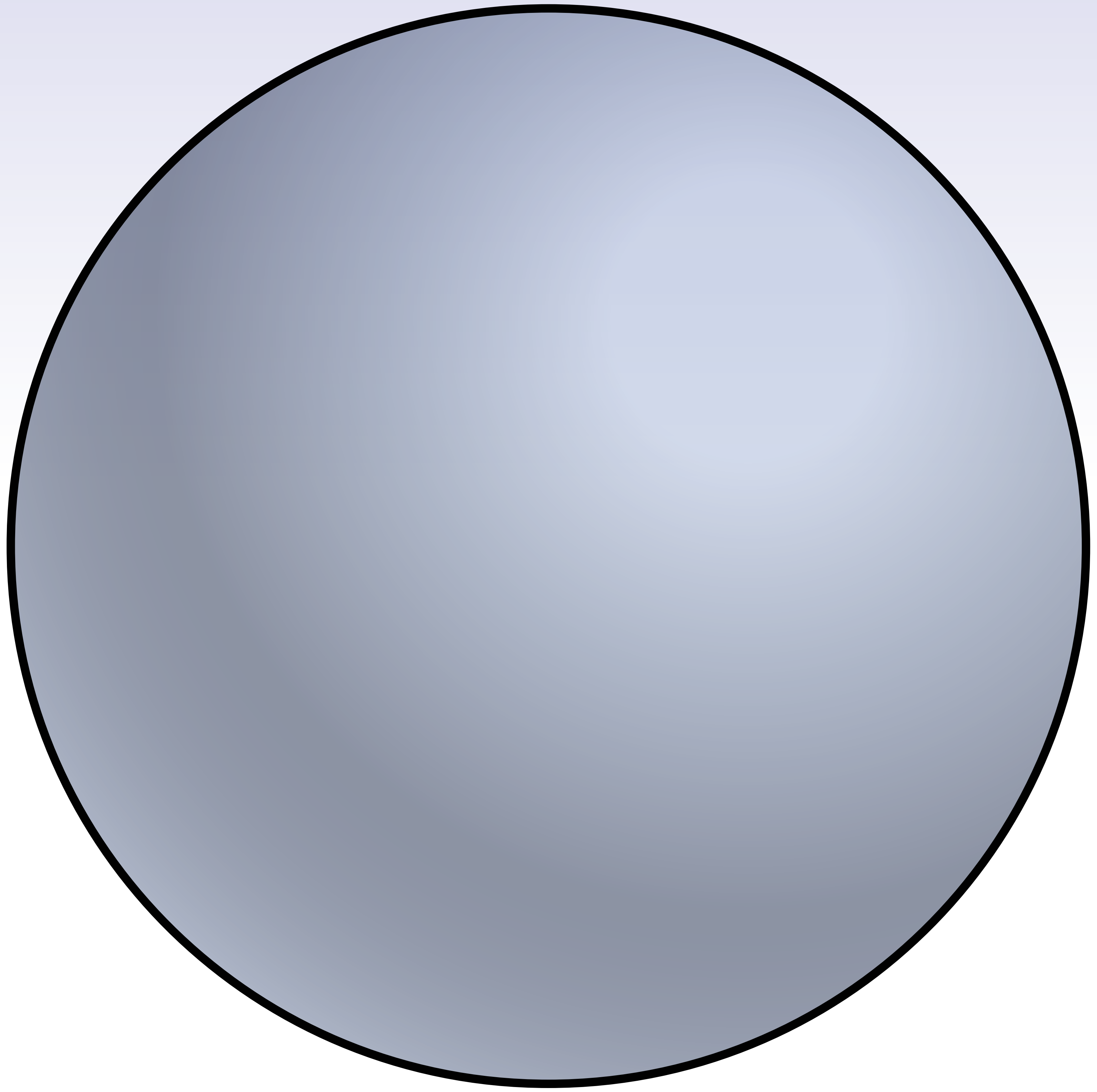


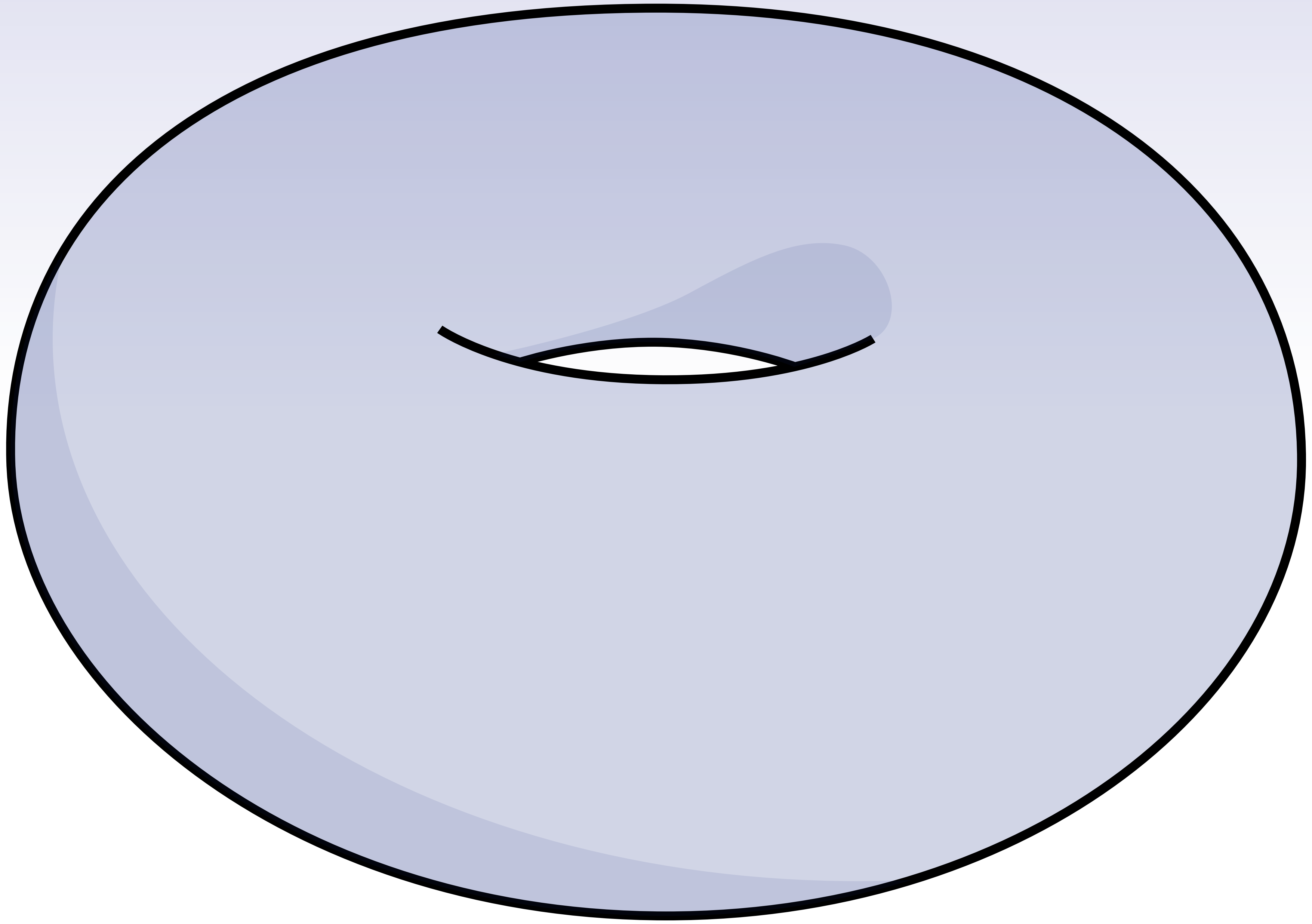
Homology on Surfaces



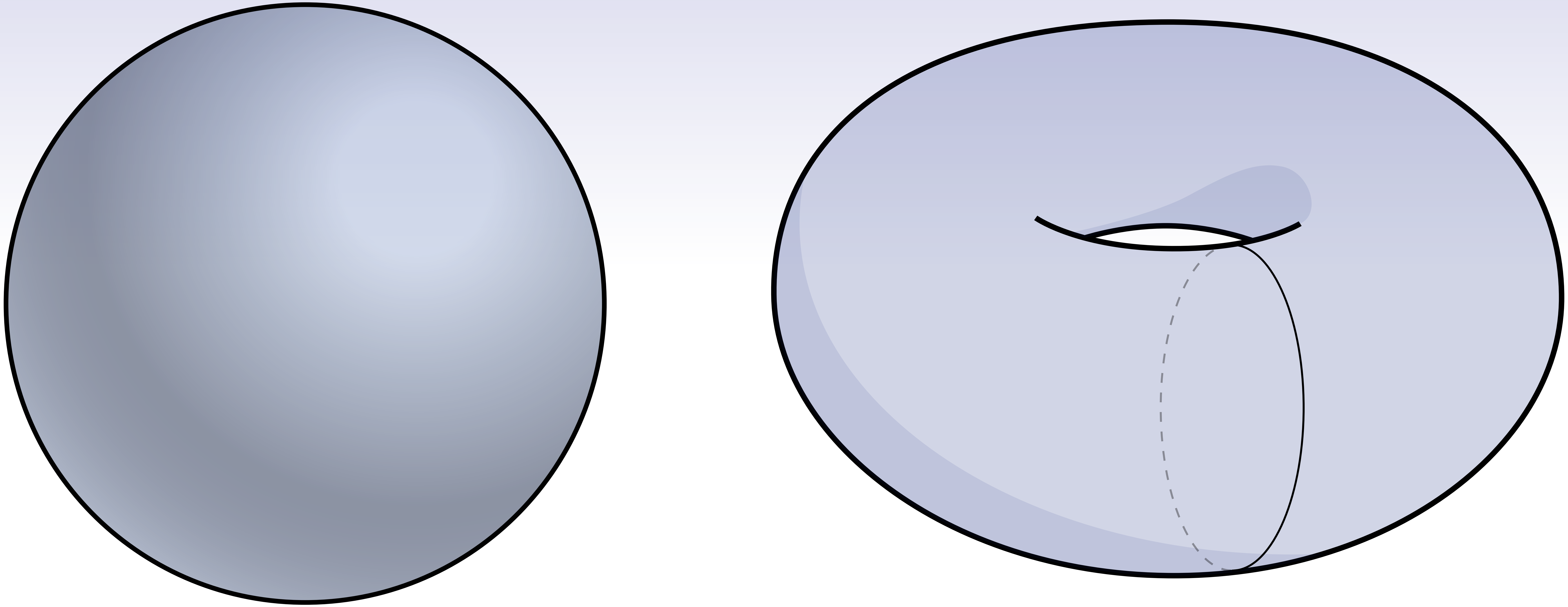


Homology on Surfaces

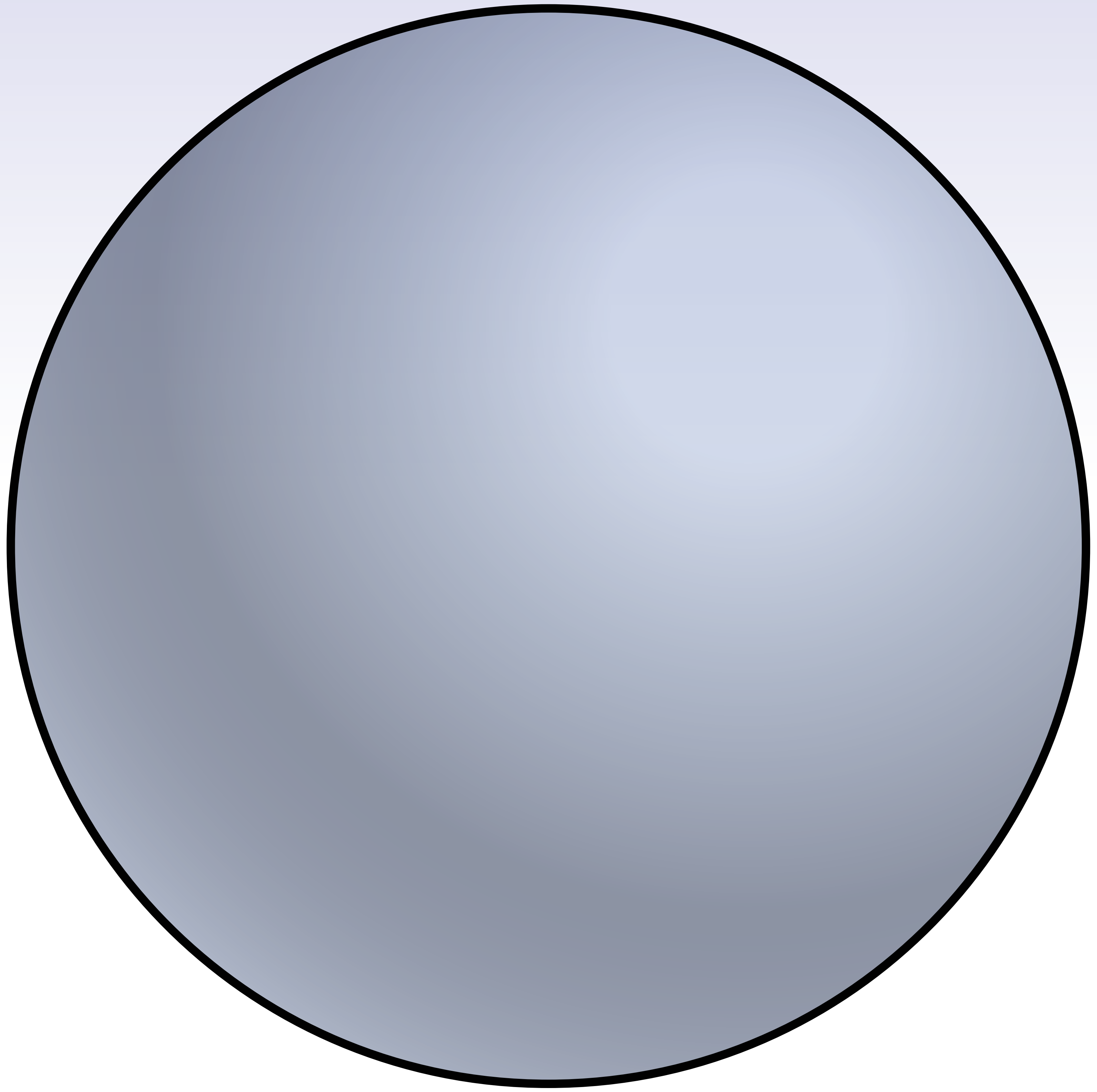


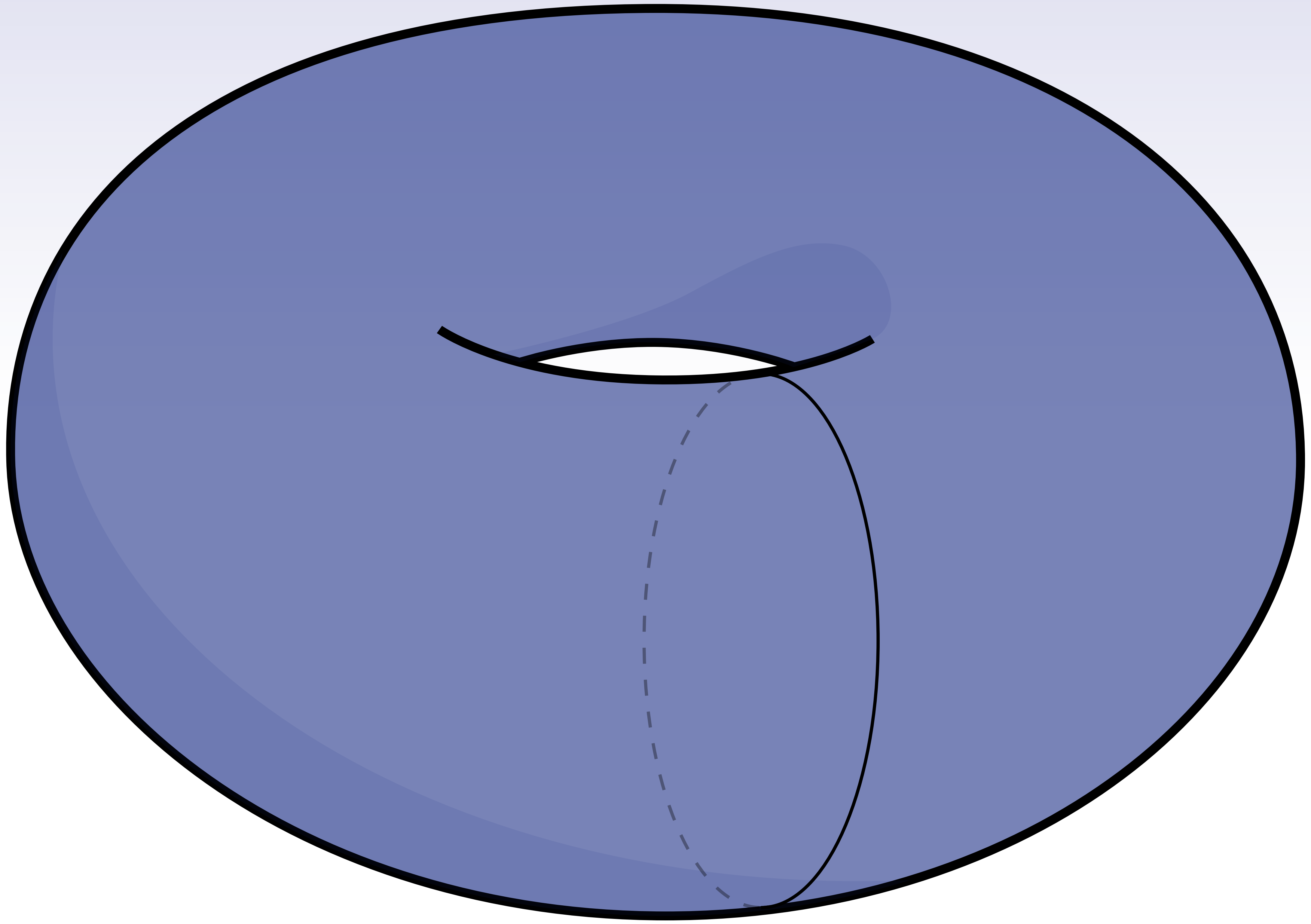


Homology on Surfaces

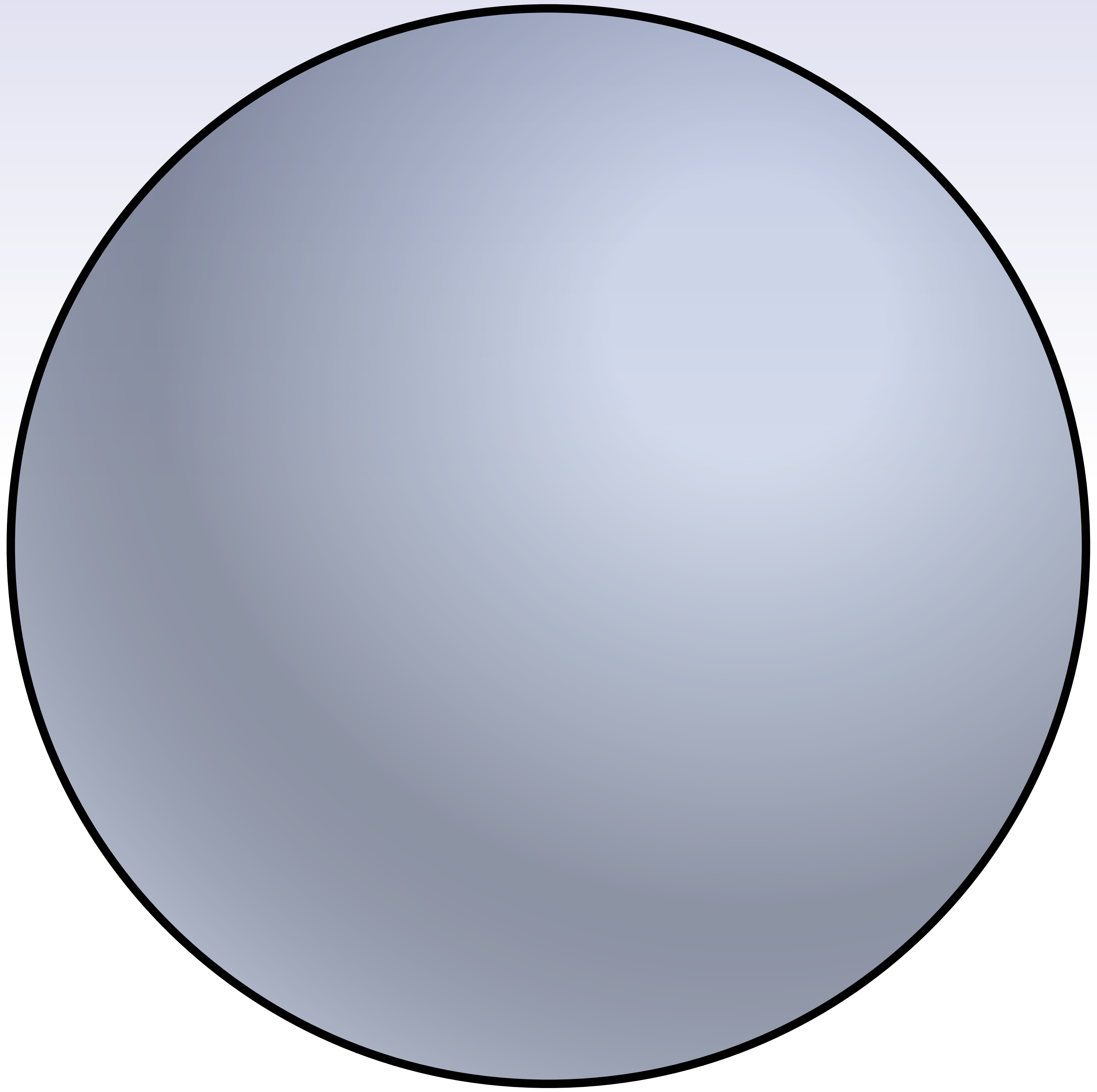


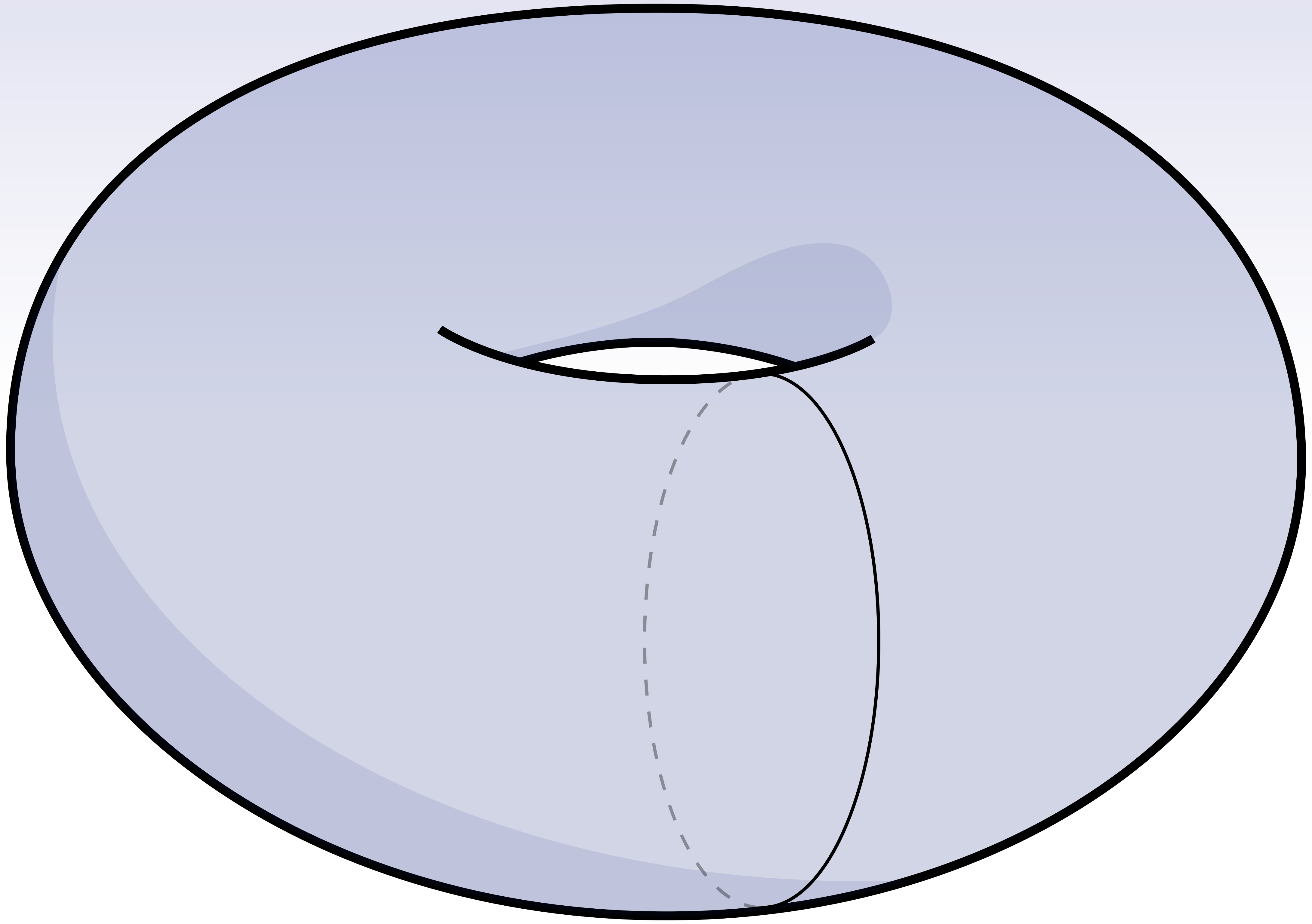
Homology on Surfaces



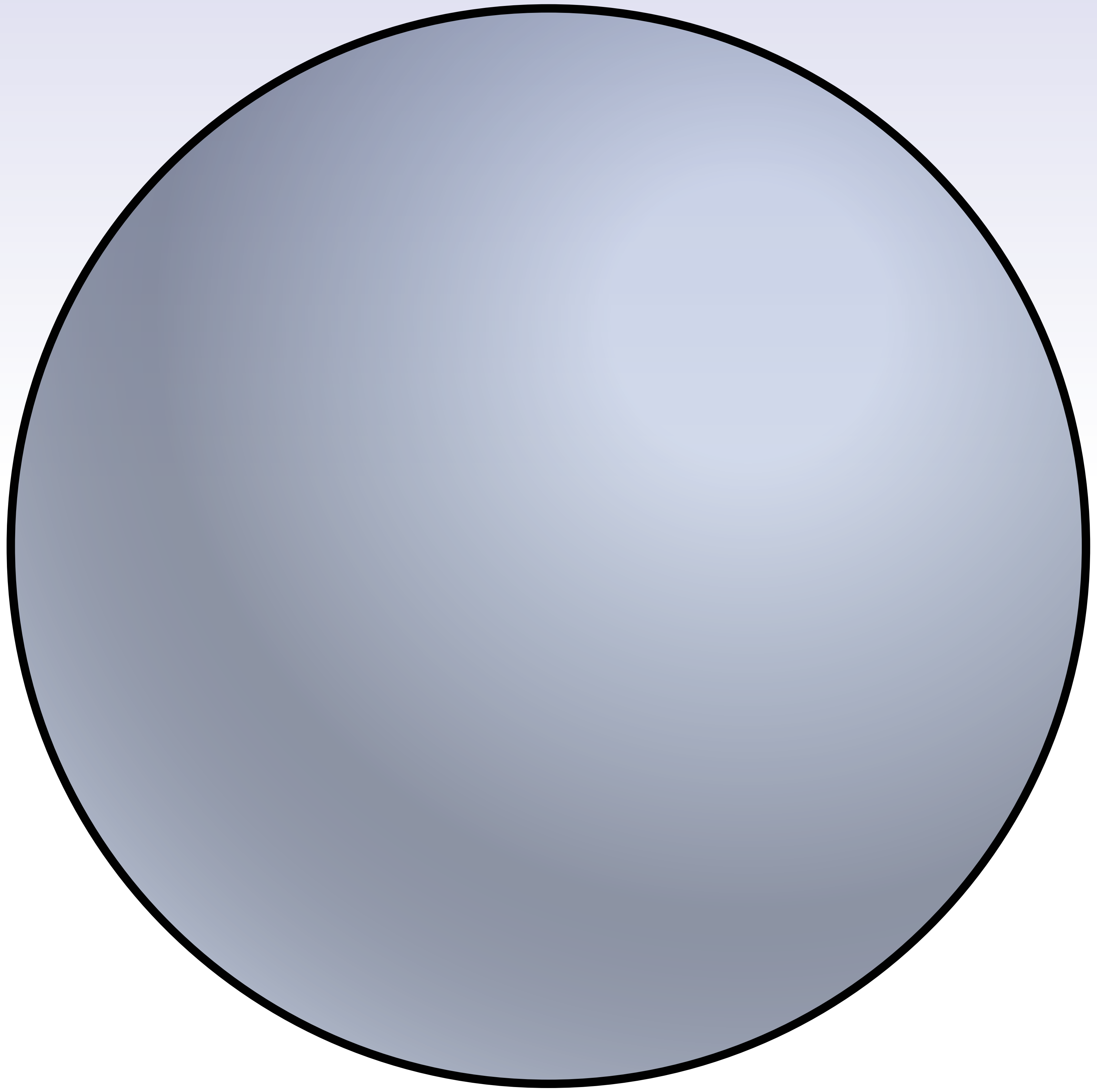


Homology on Surfaces

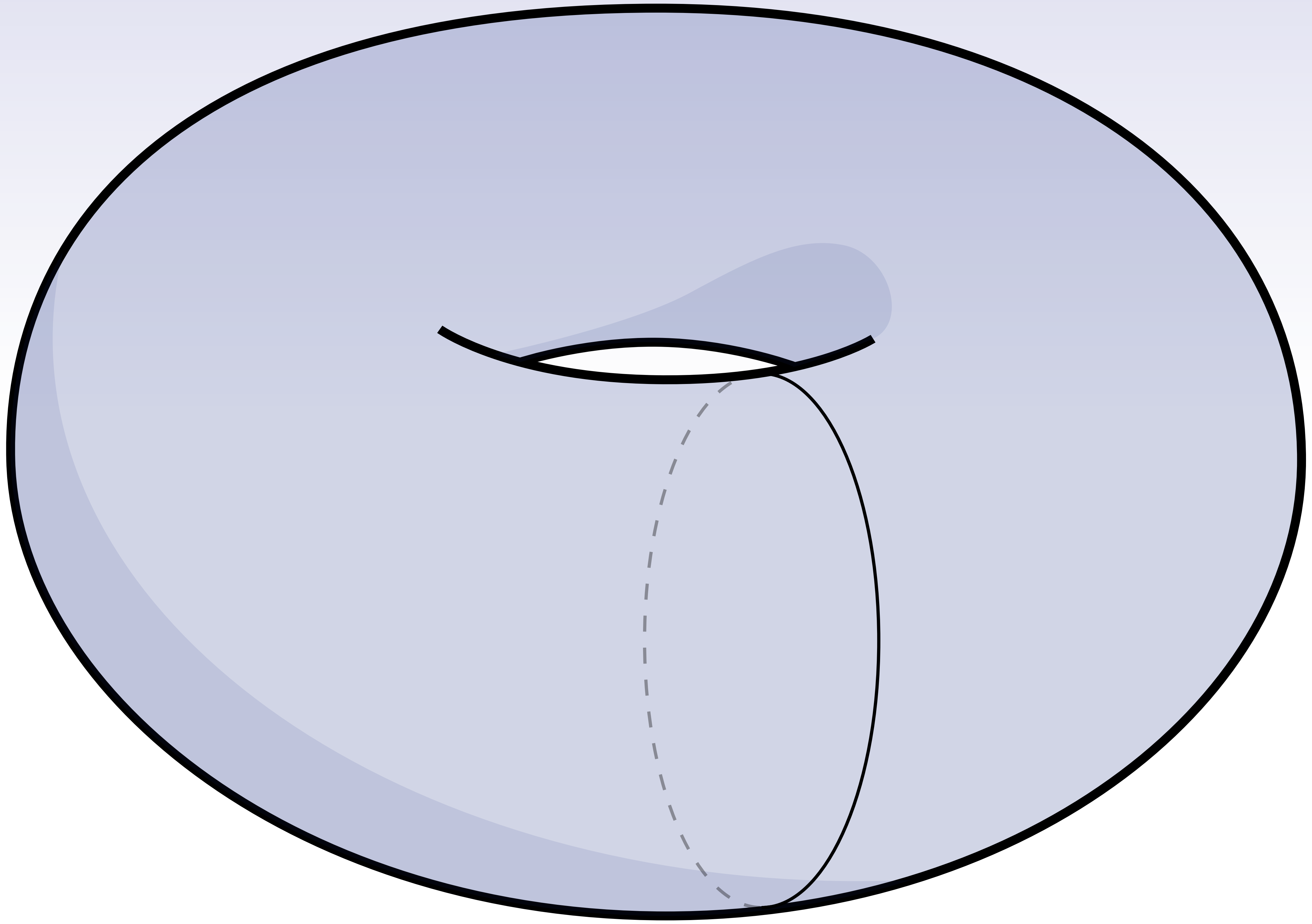




Homology on Surfaces

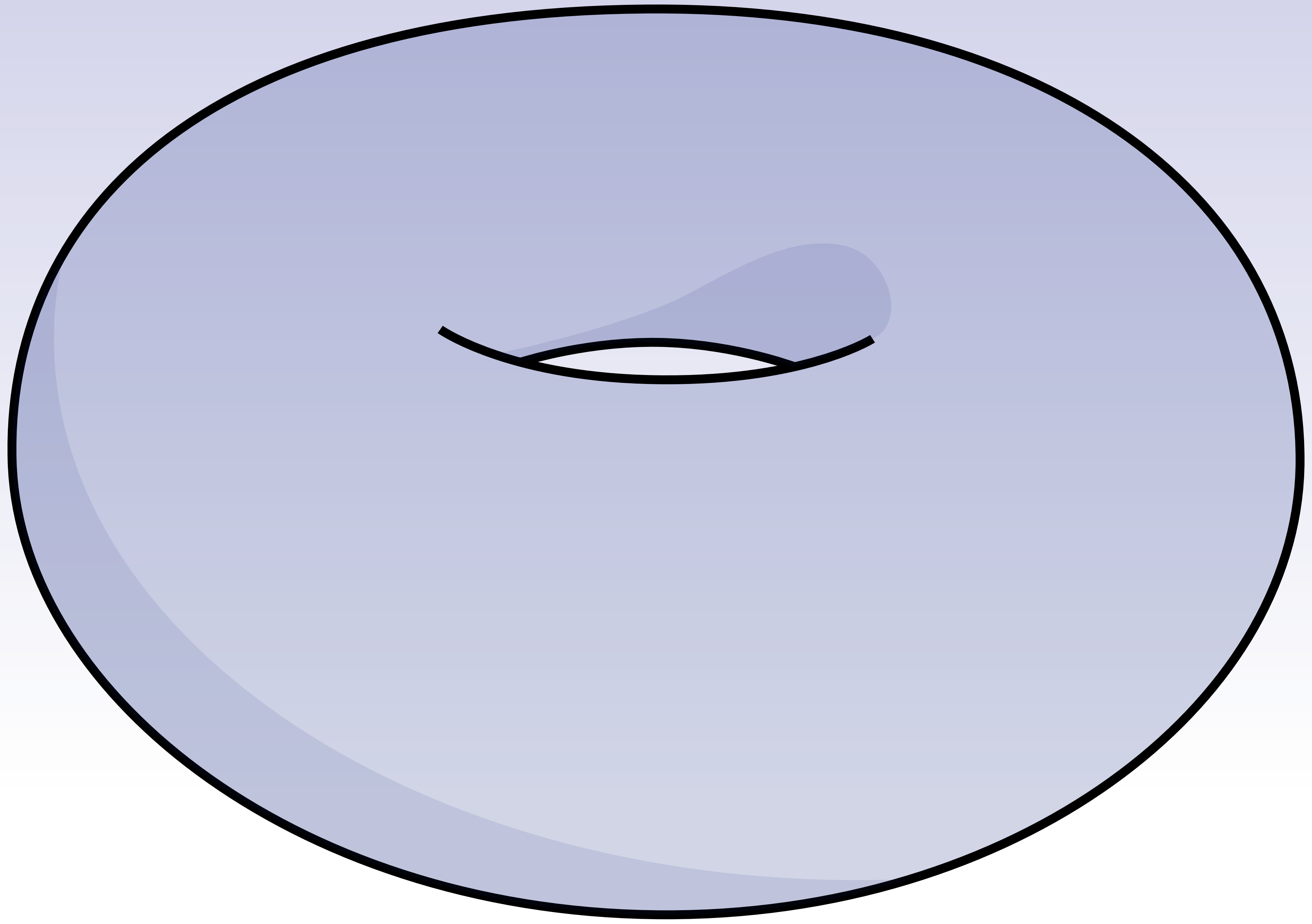


simply connected

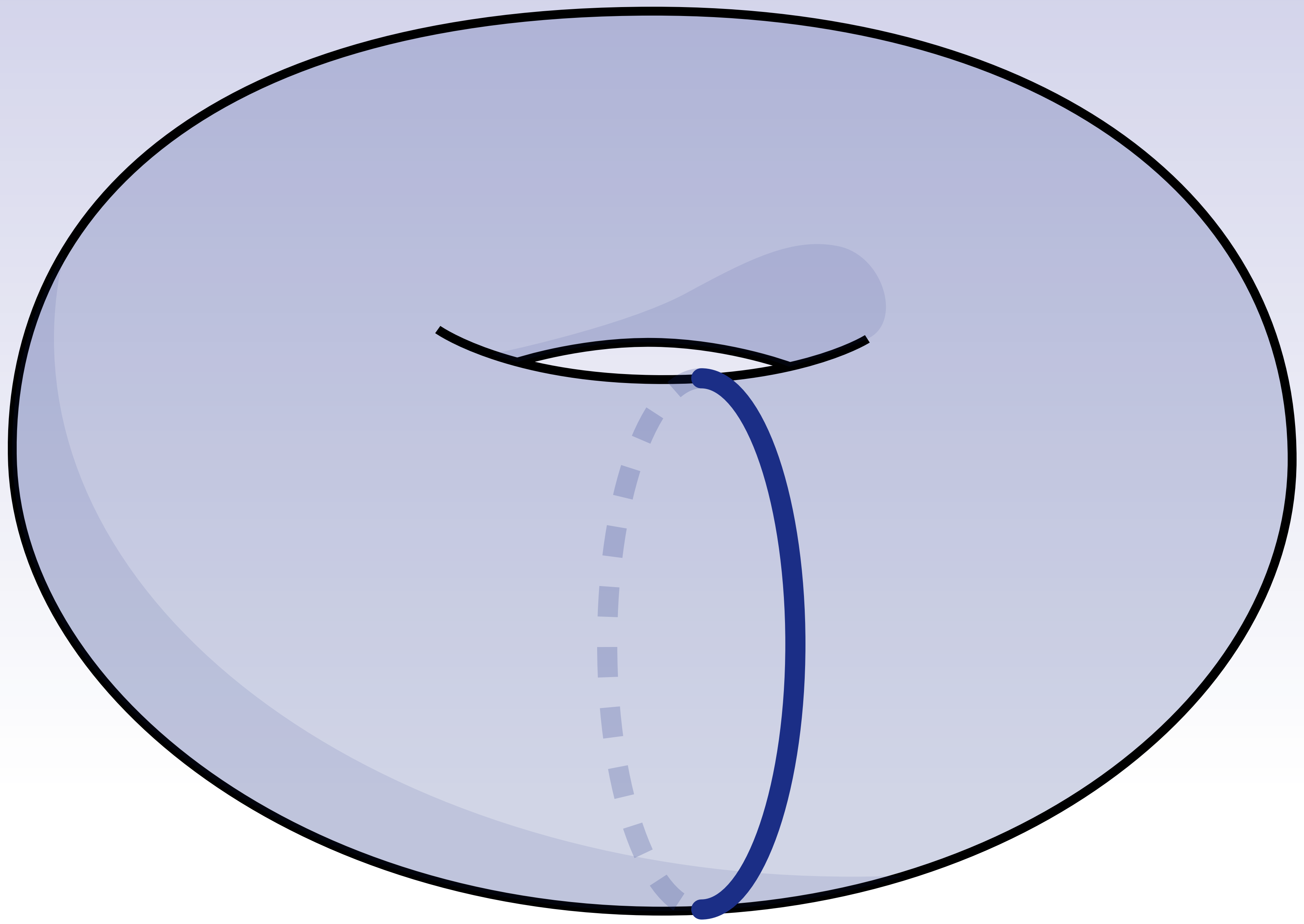


nonsimply connected

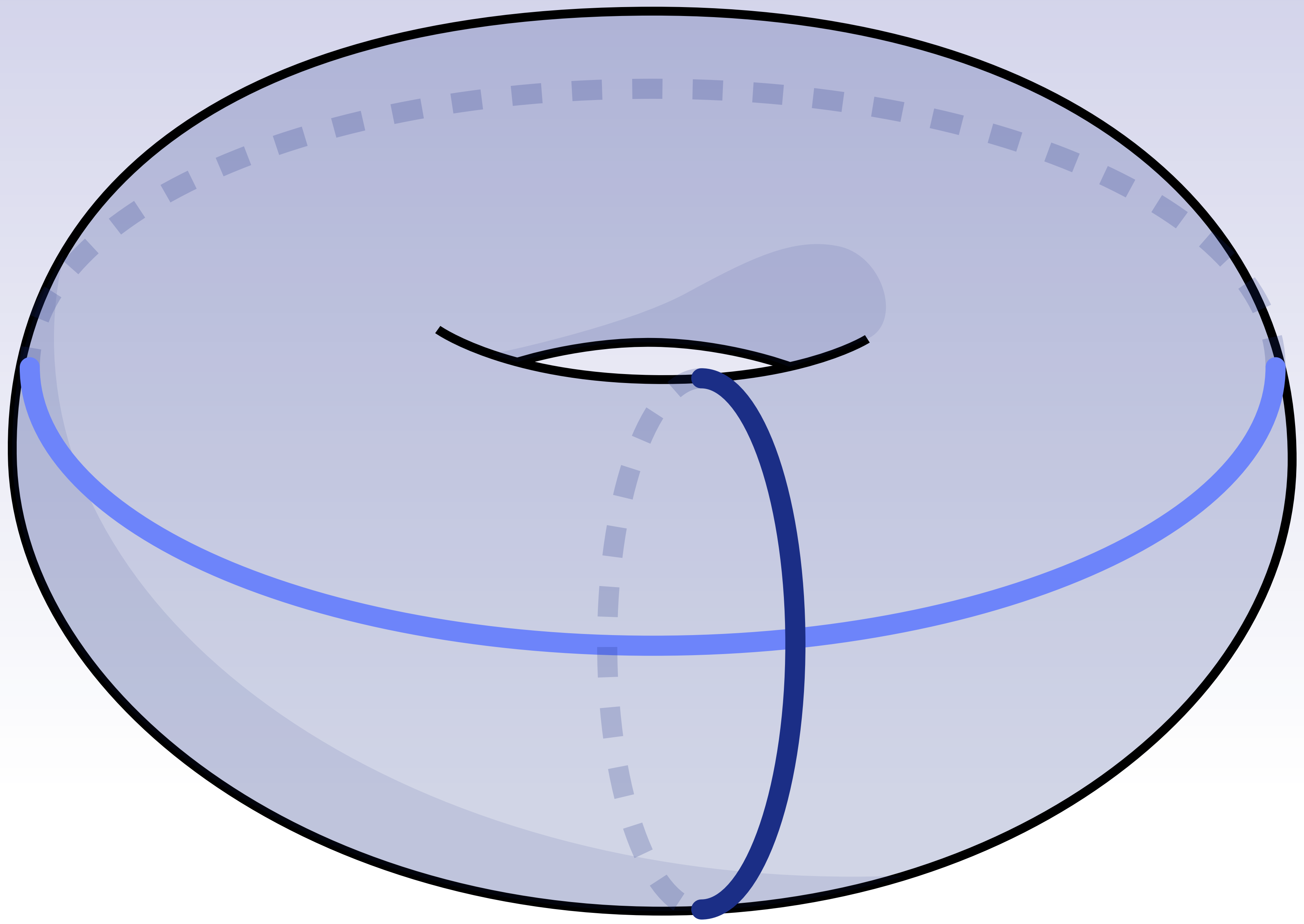
Homology Generators



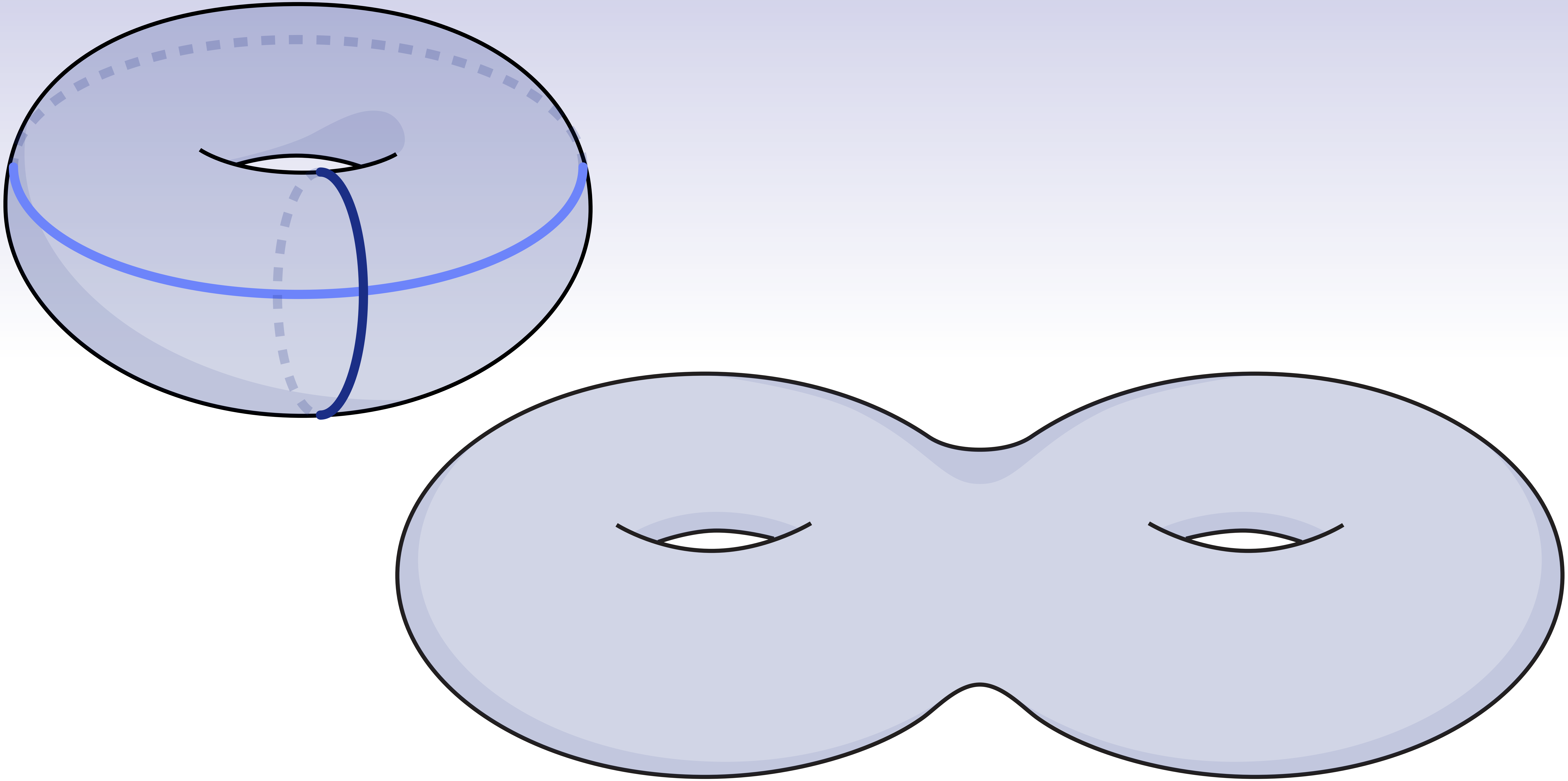
Homology Generators



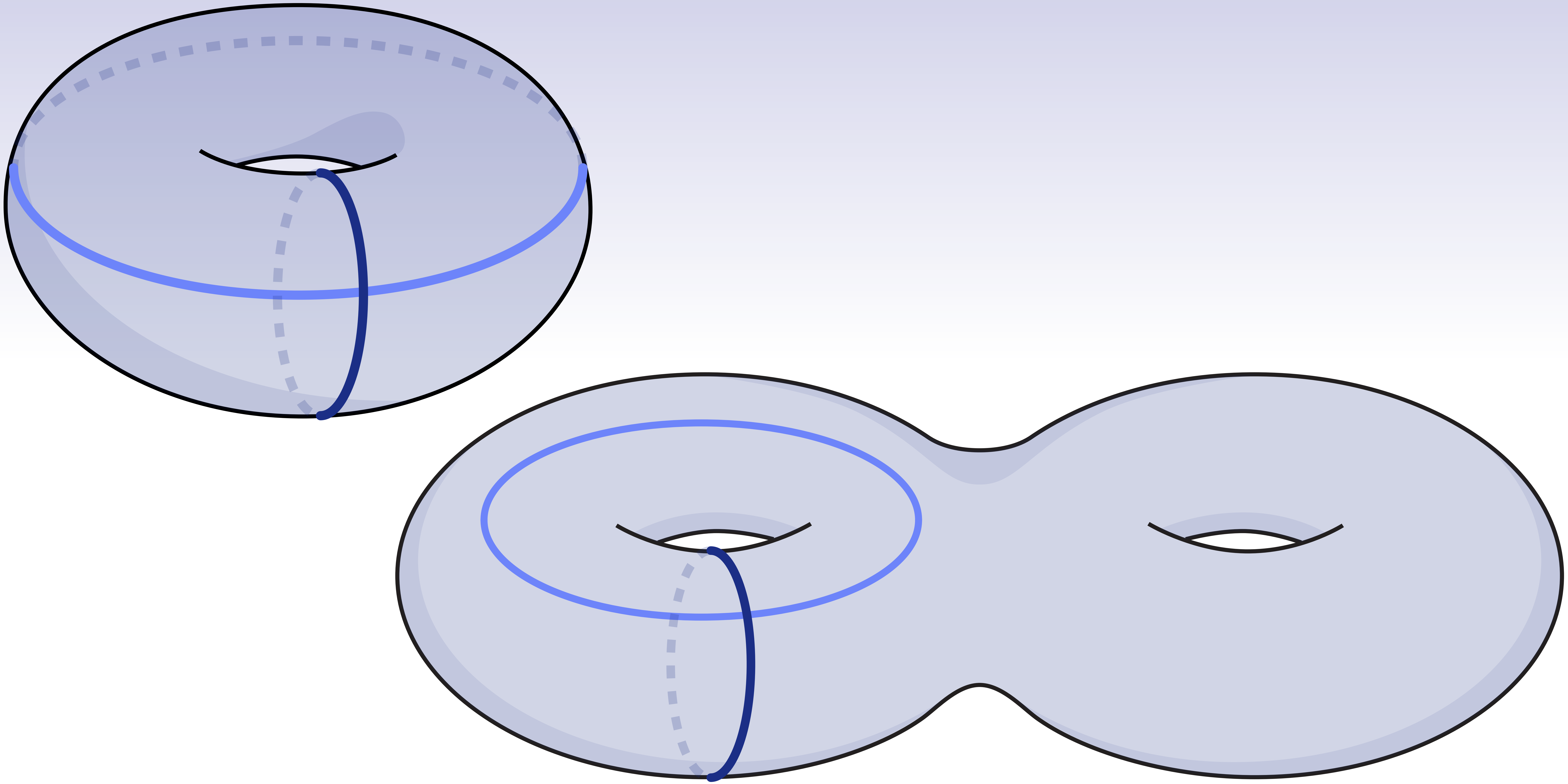
Homology Generators



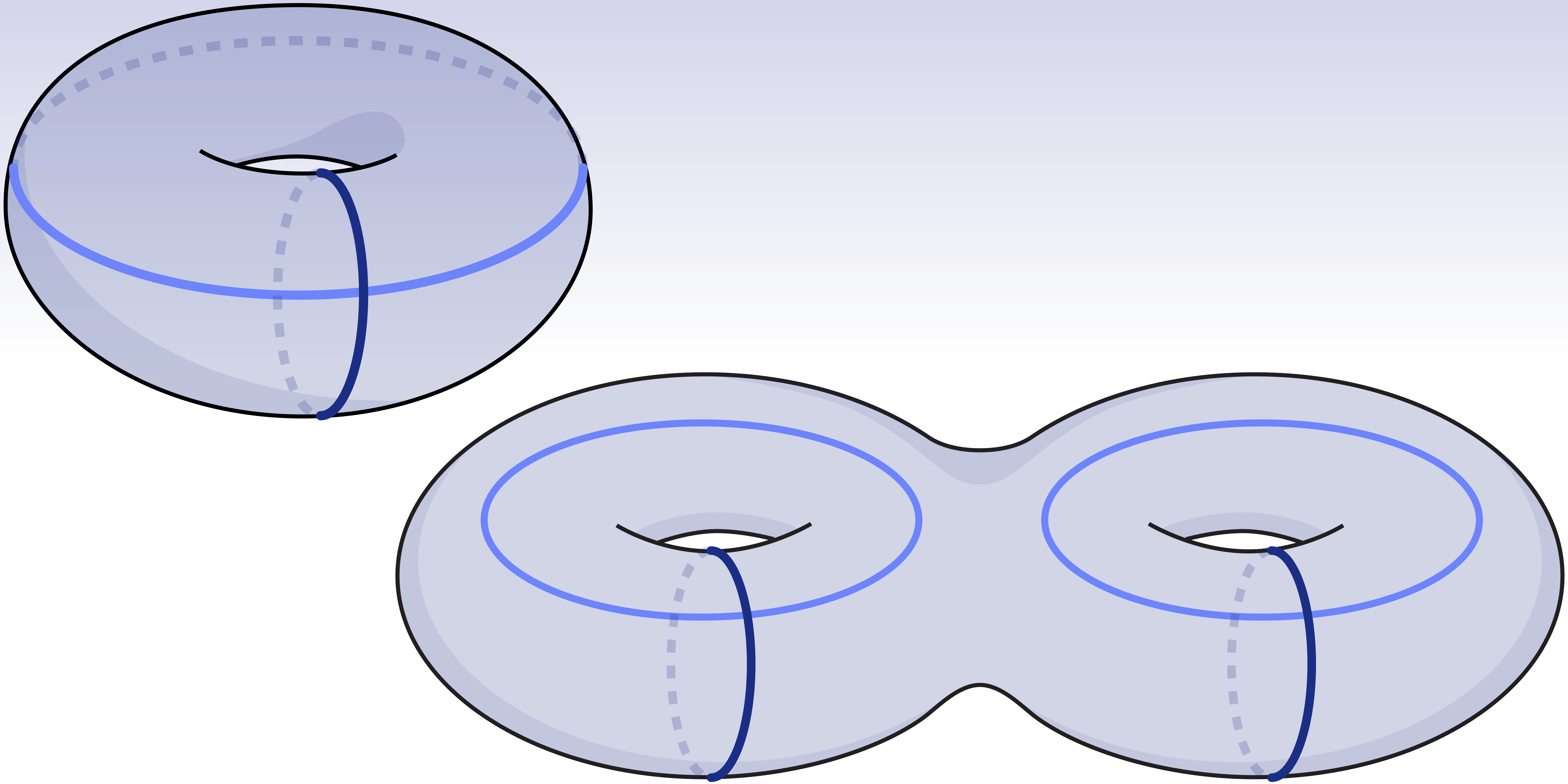
Homology Generators



Homology Generators



Homology Generators



Tree-Cotree Decomposition

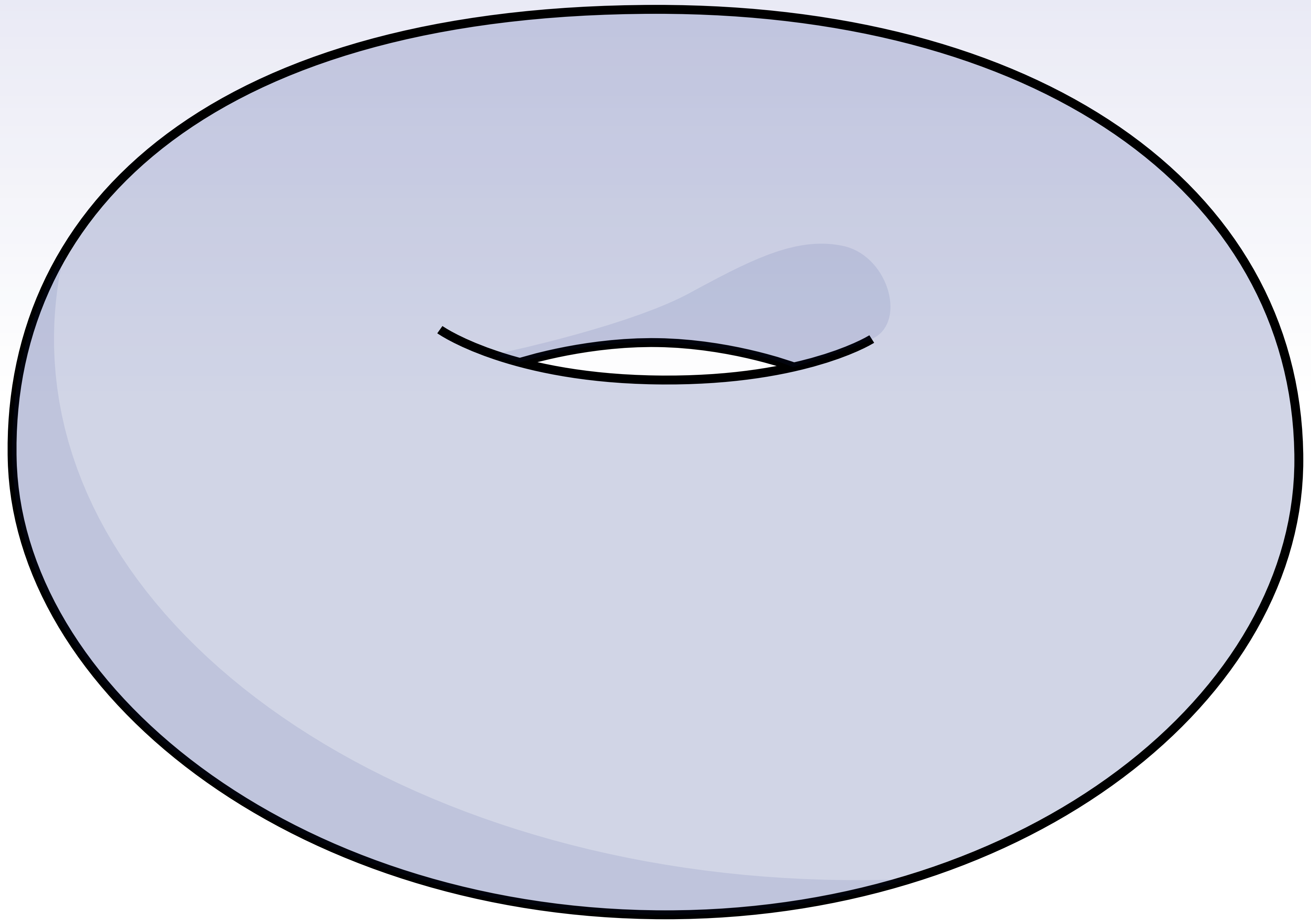
Tree-Cotree Decomposition

Tree-Cotree Decomposition

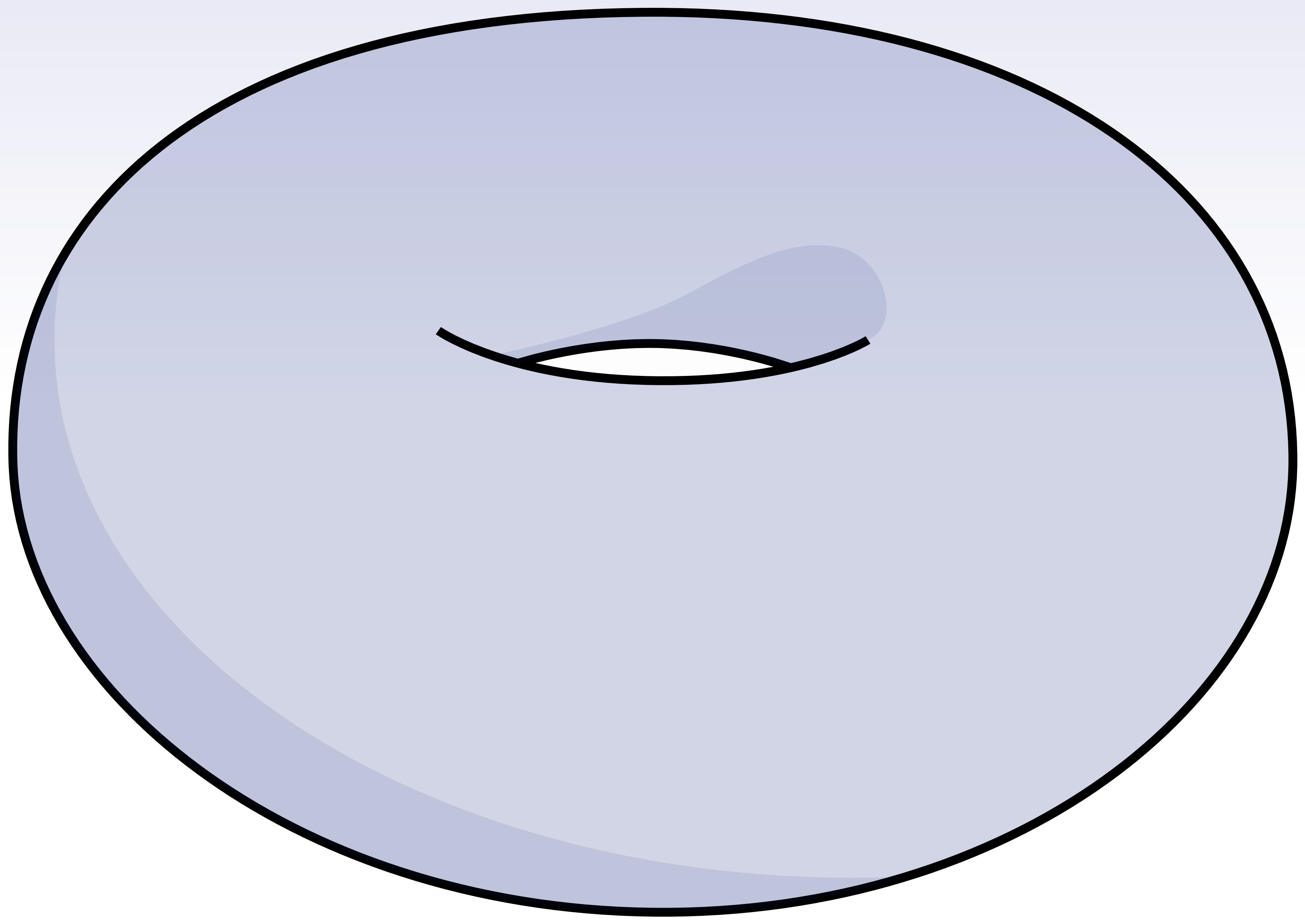
Tree-Cotree Decomposition

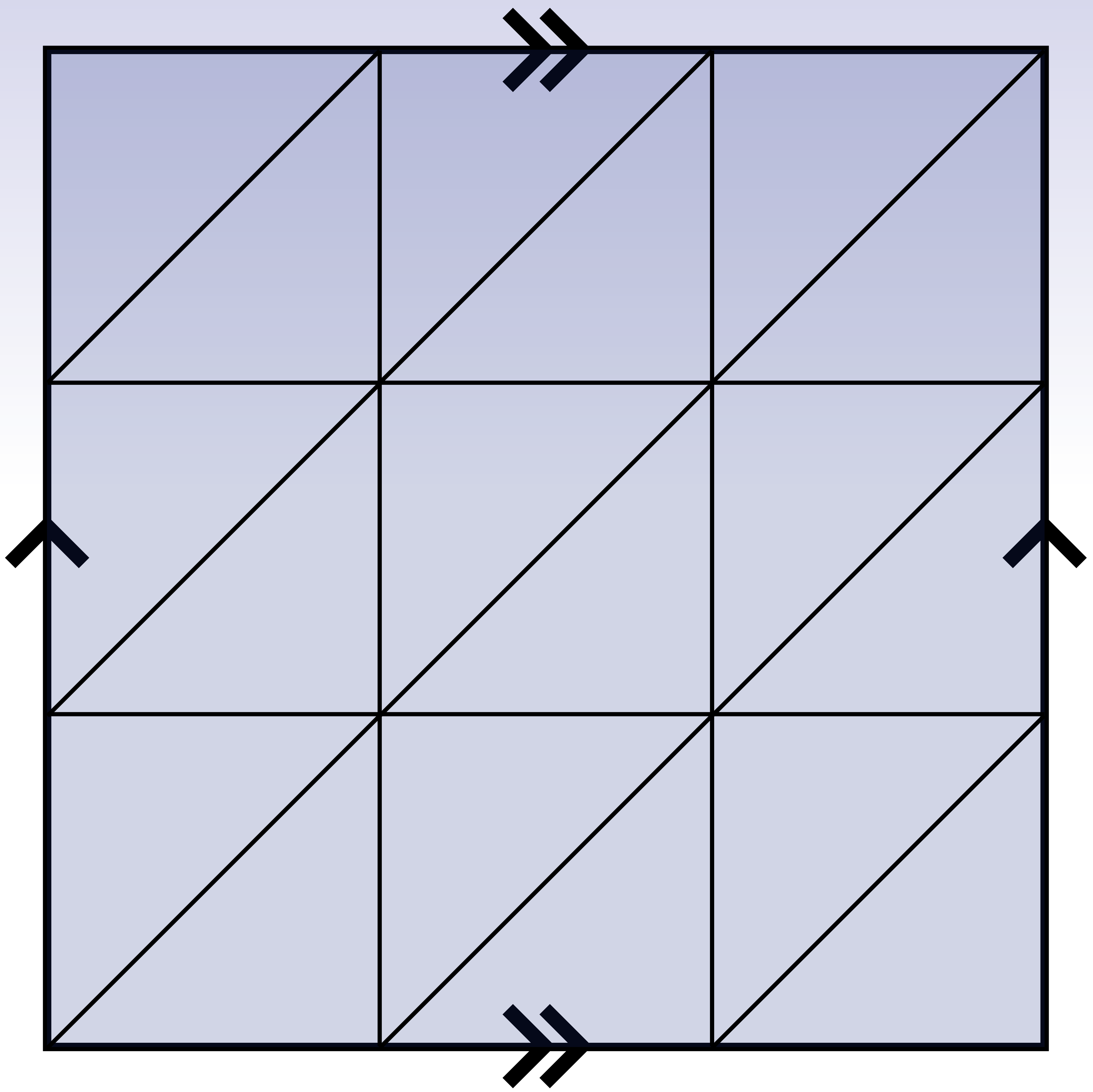
Example: Generators on Torus

Example: Generators on Torus

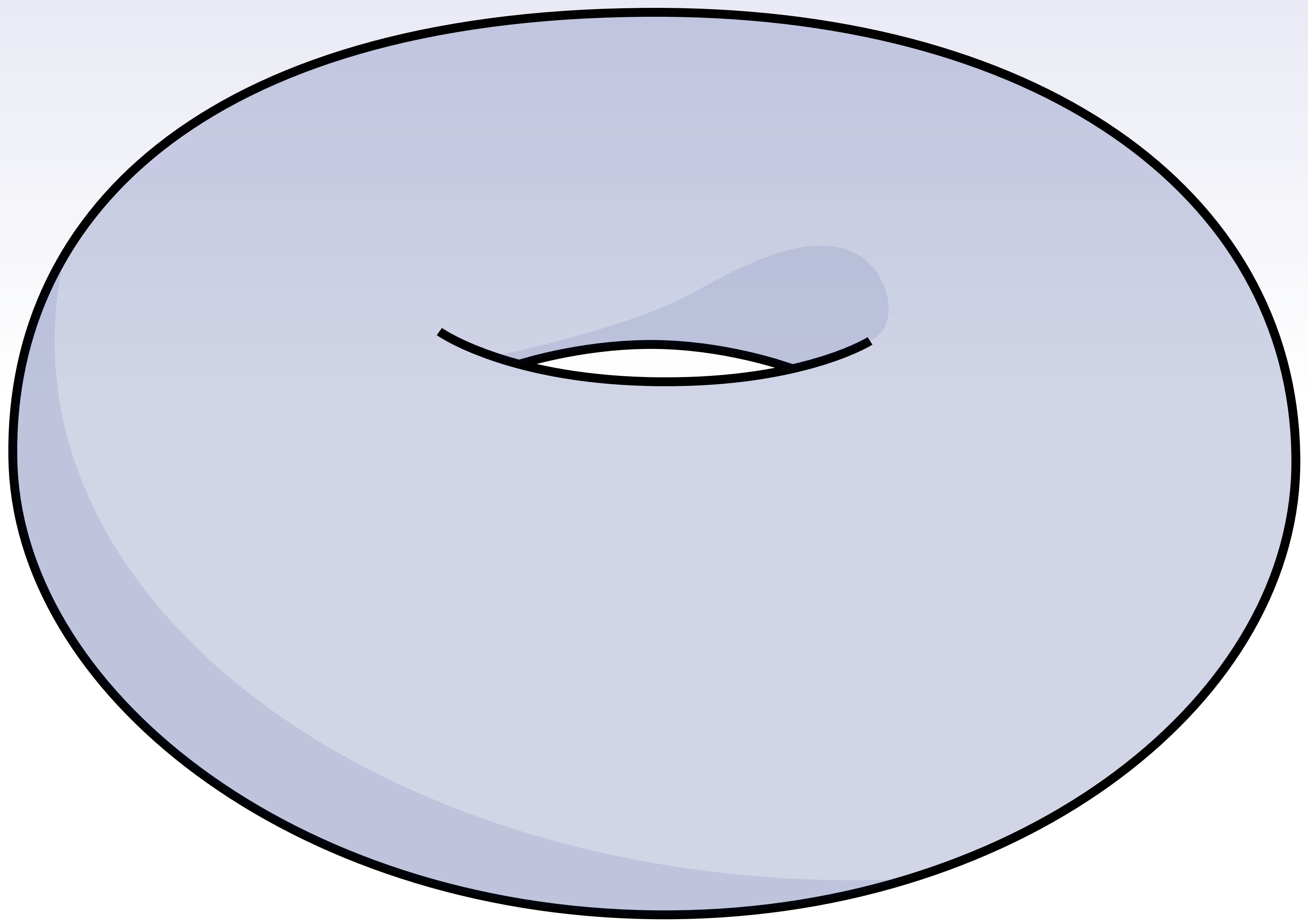


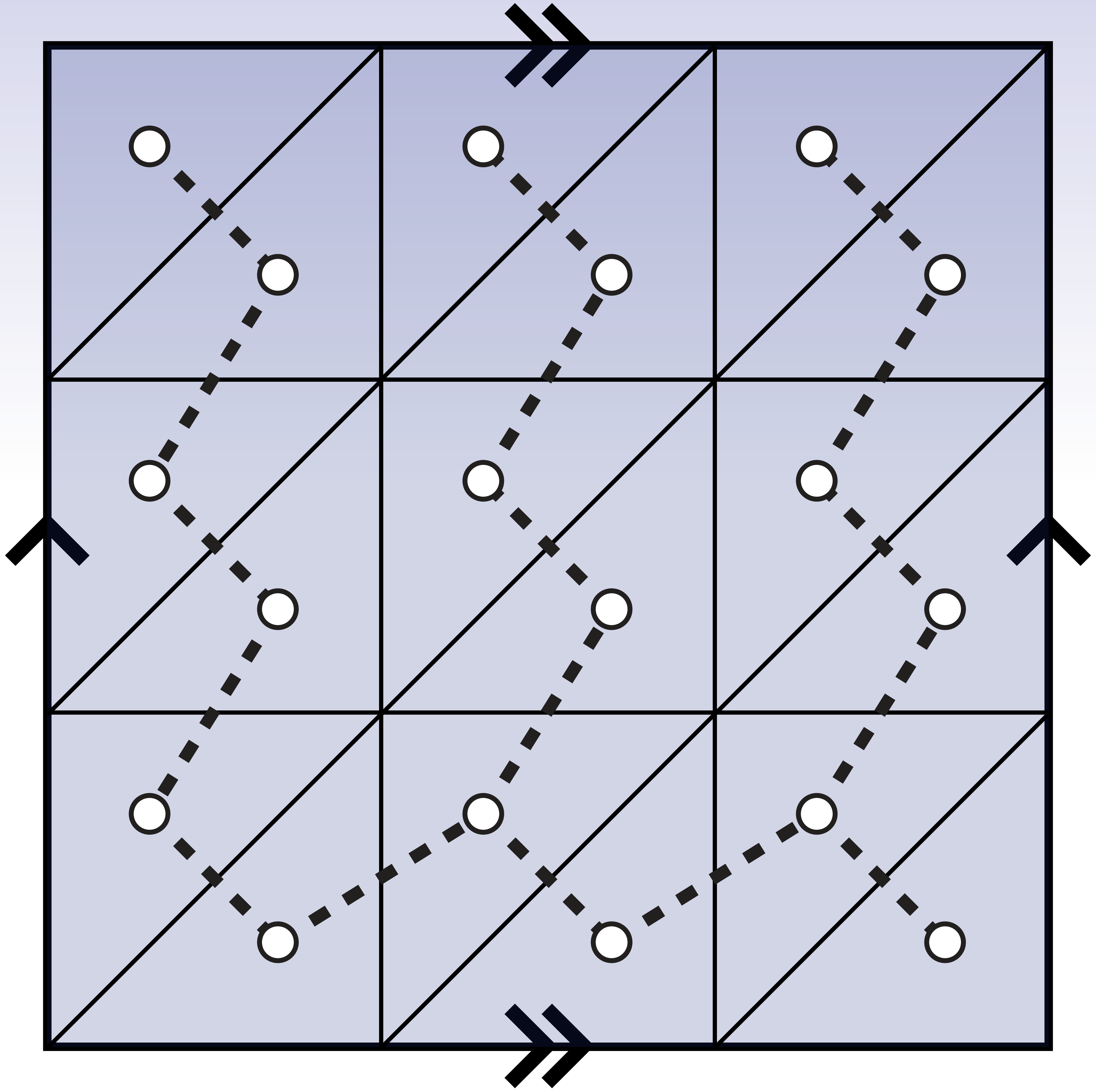
Example: Generators on Torus



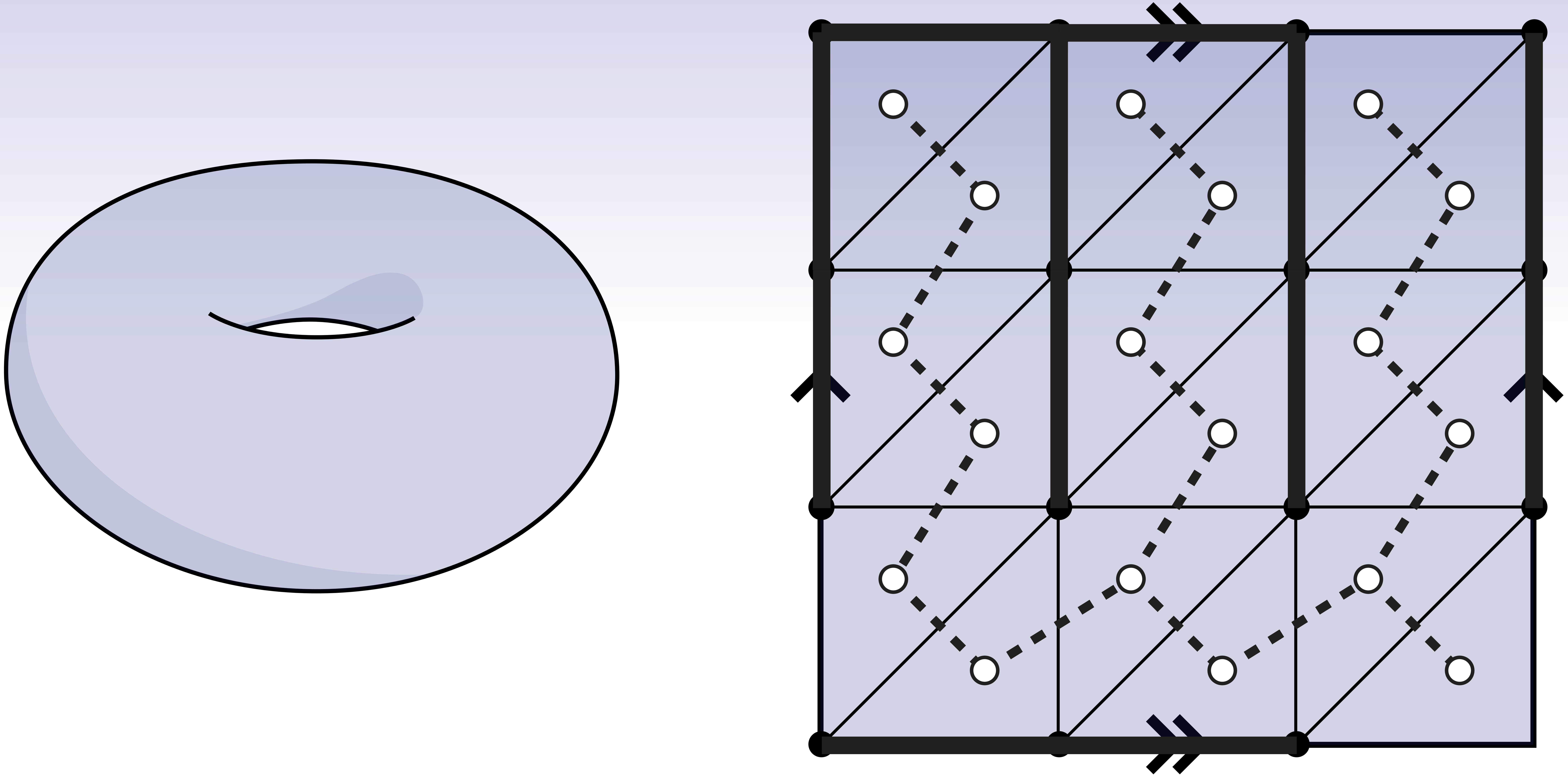


Example: Generators on Torus

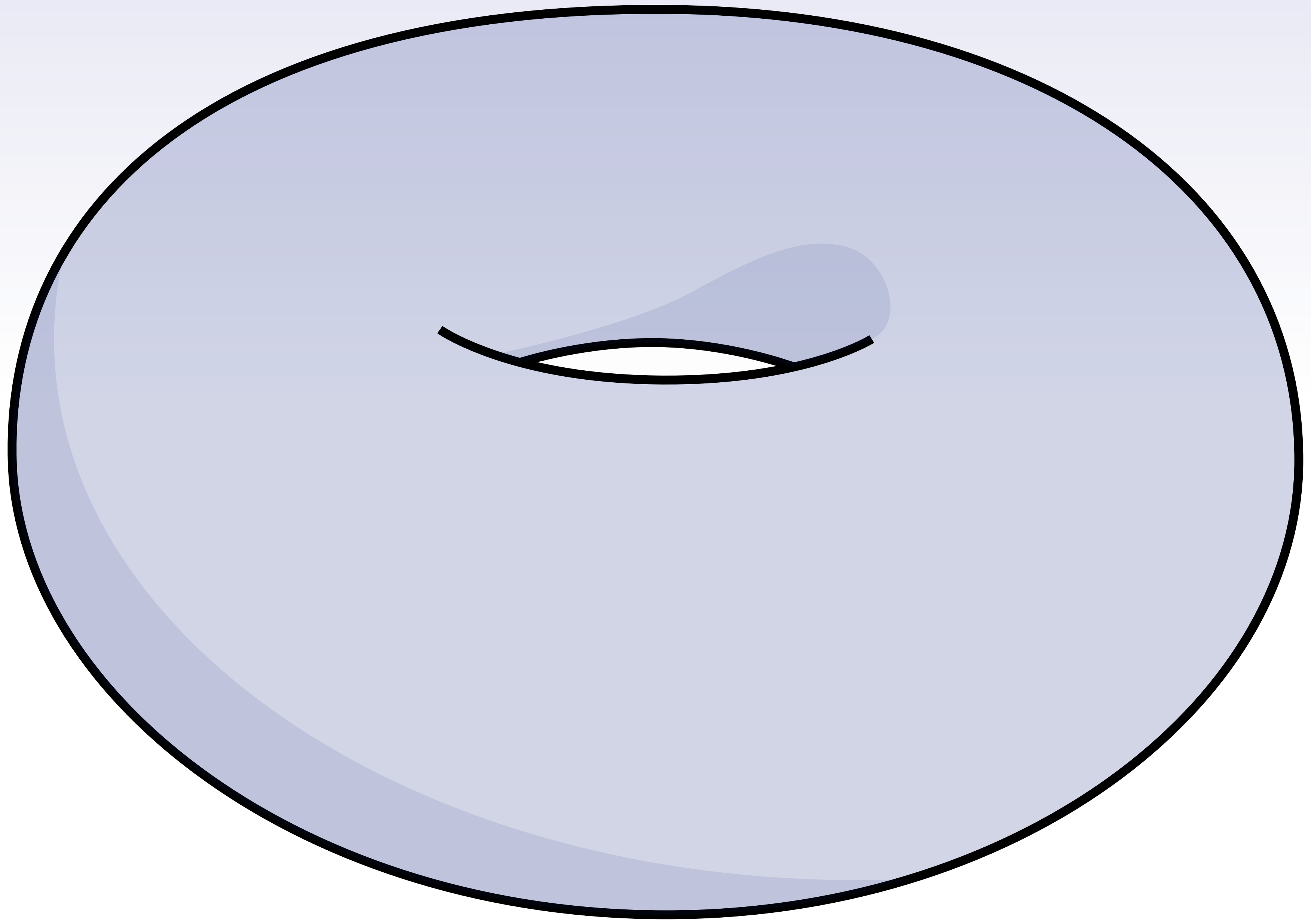


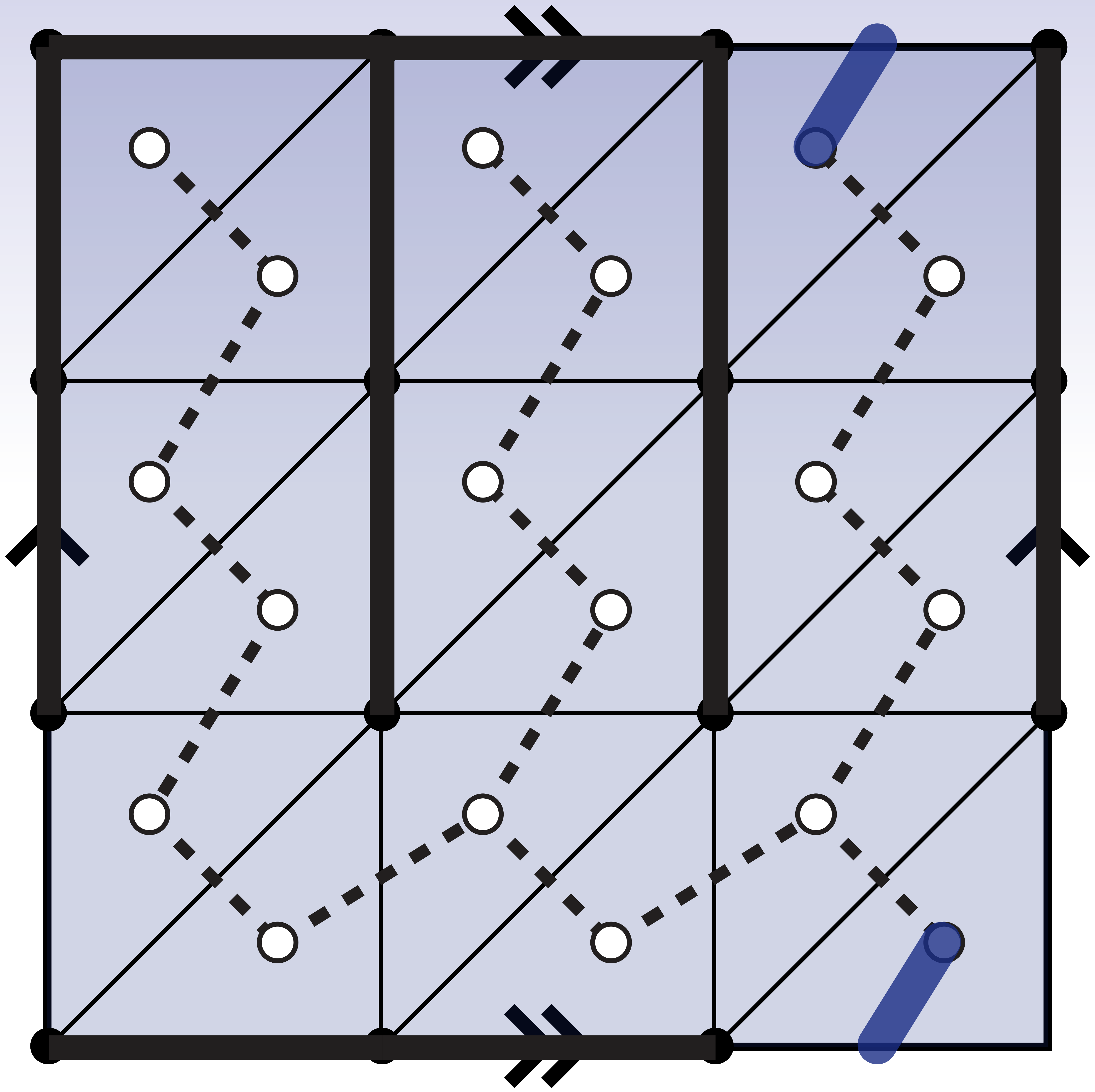


Example: Generators on Torus

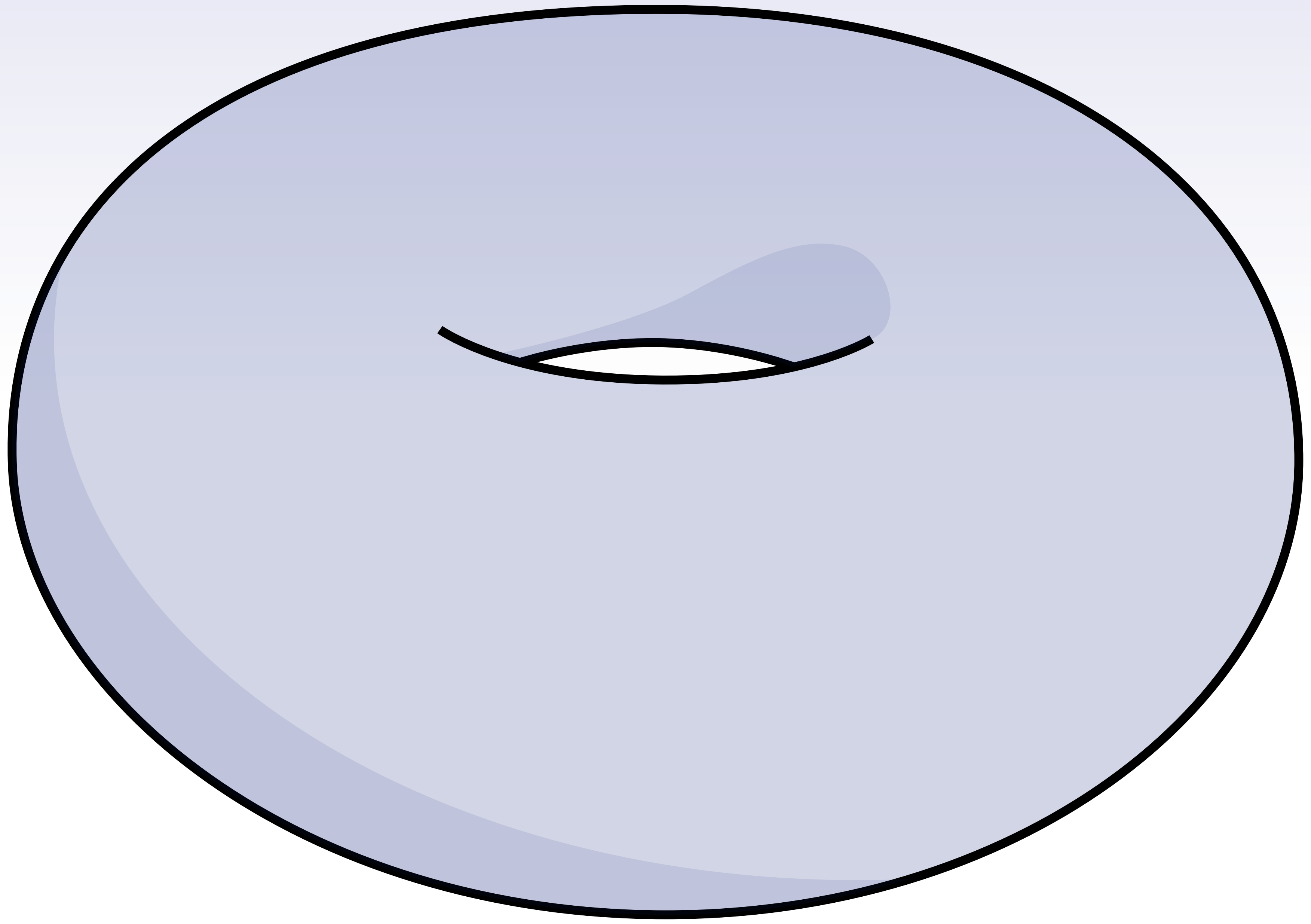


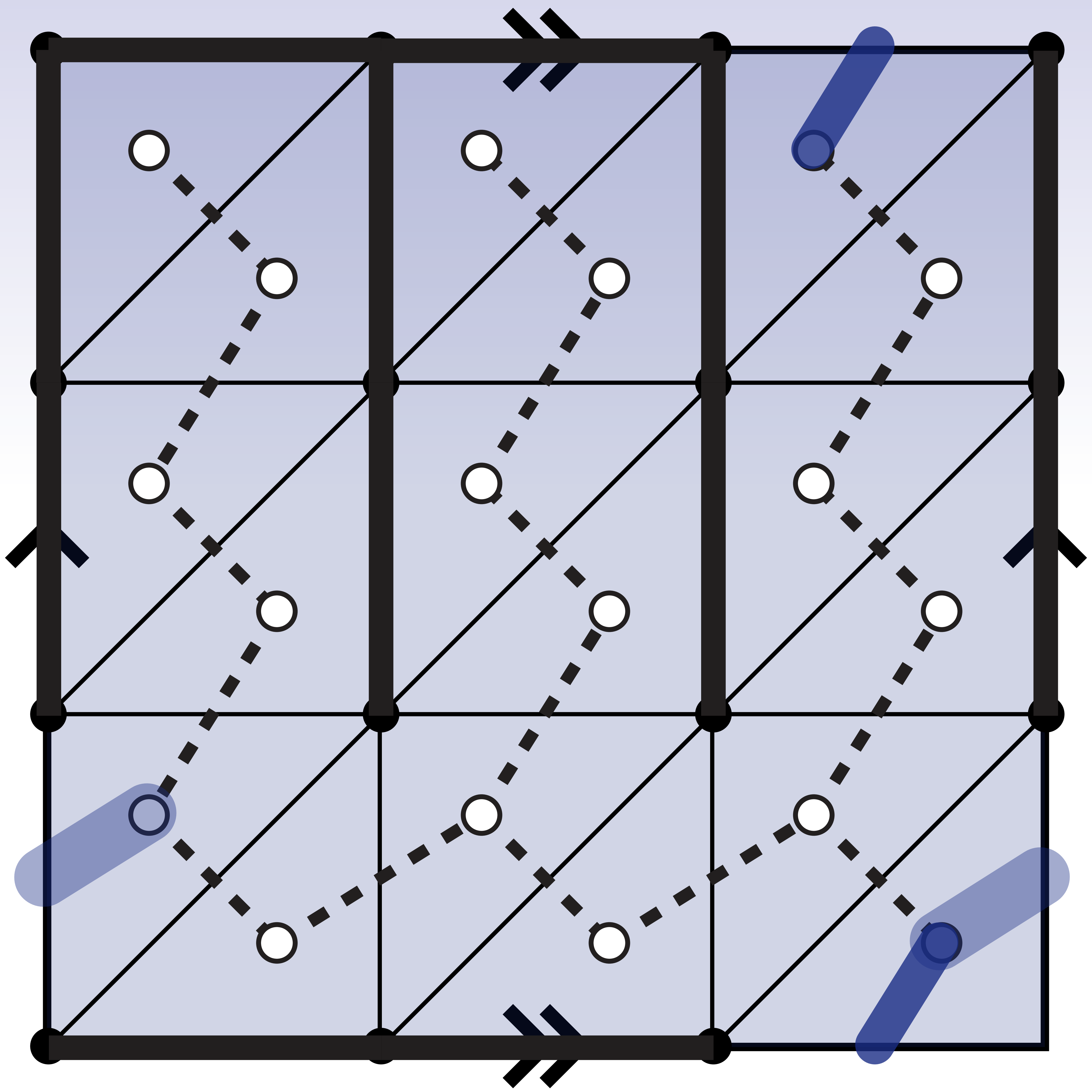
Example: Generators on Torus



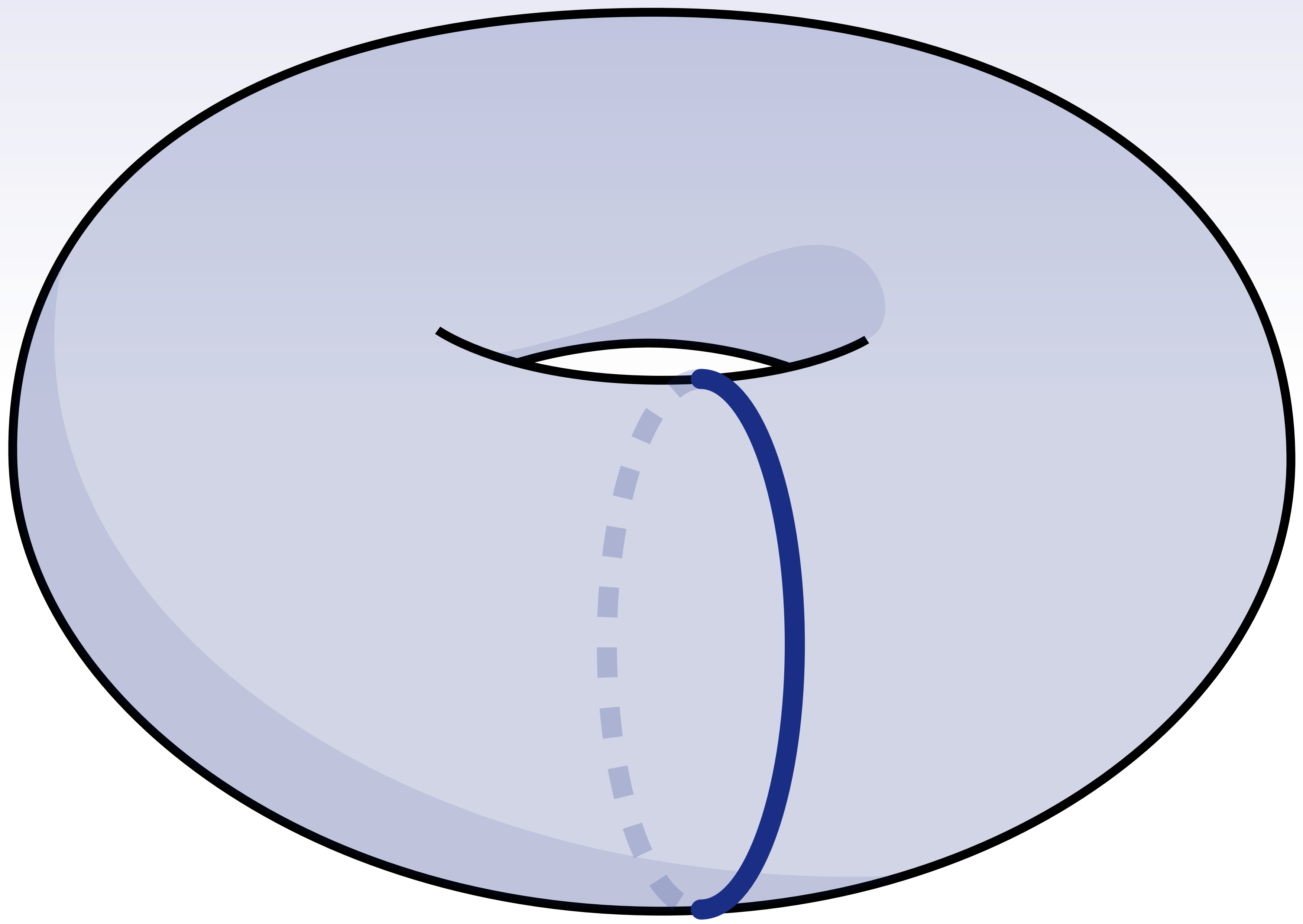


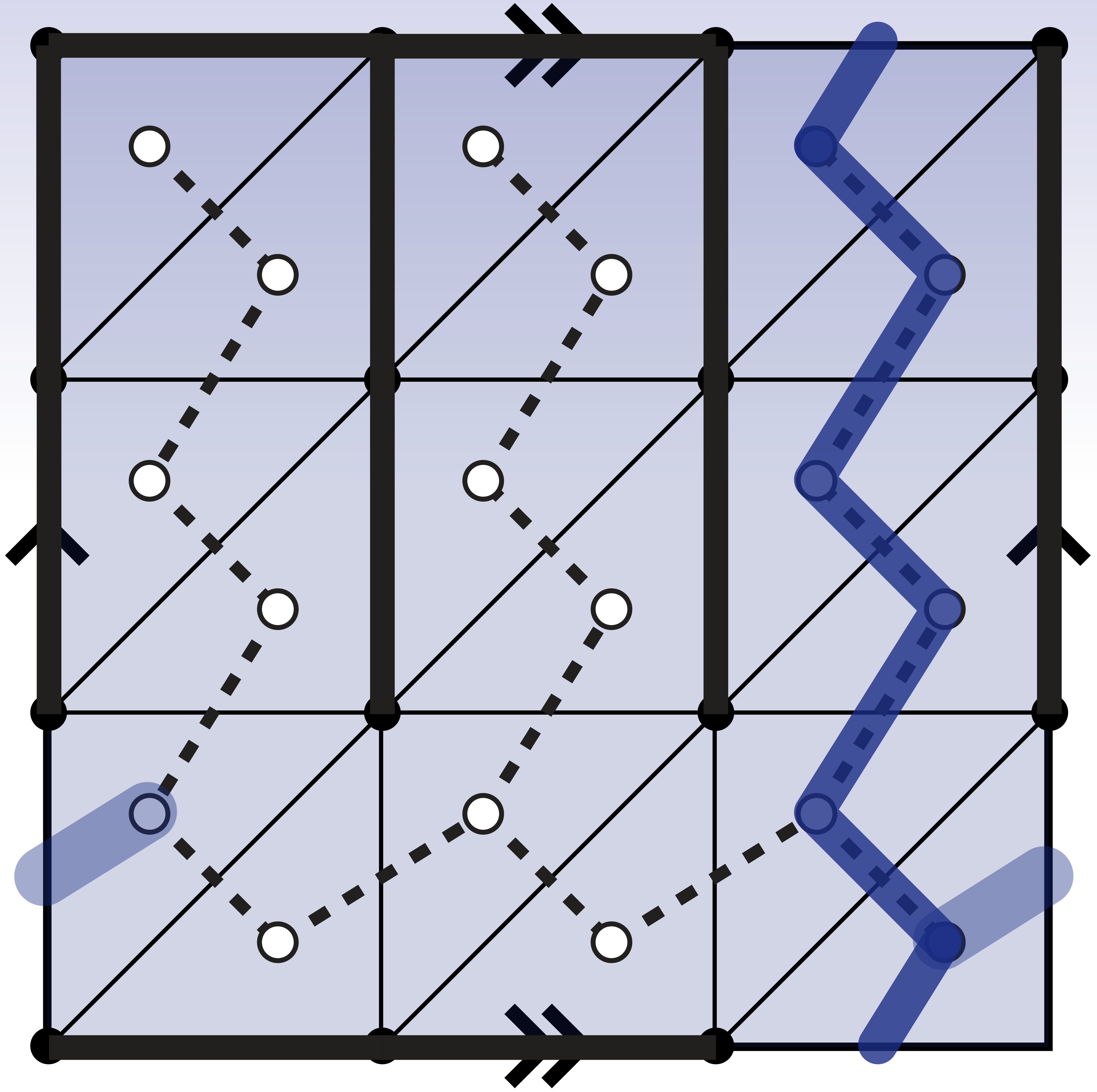
Example: Generators on Torus



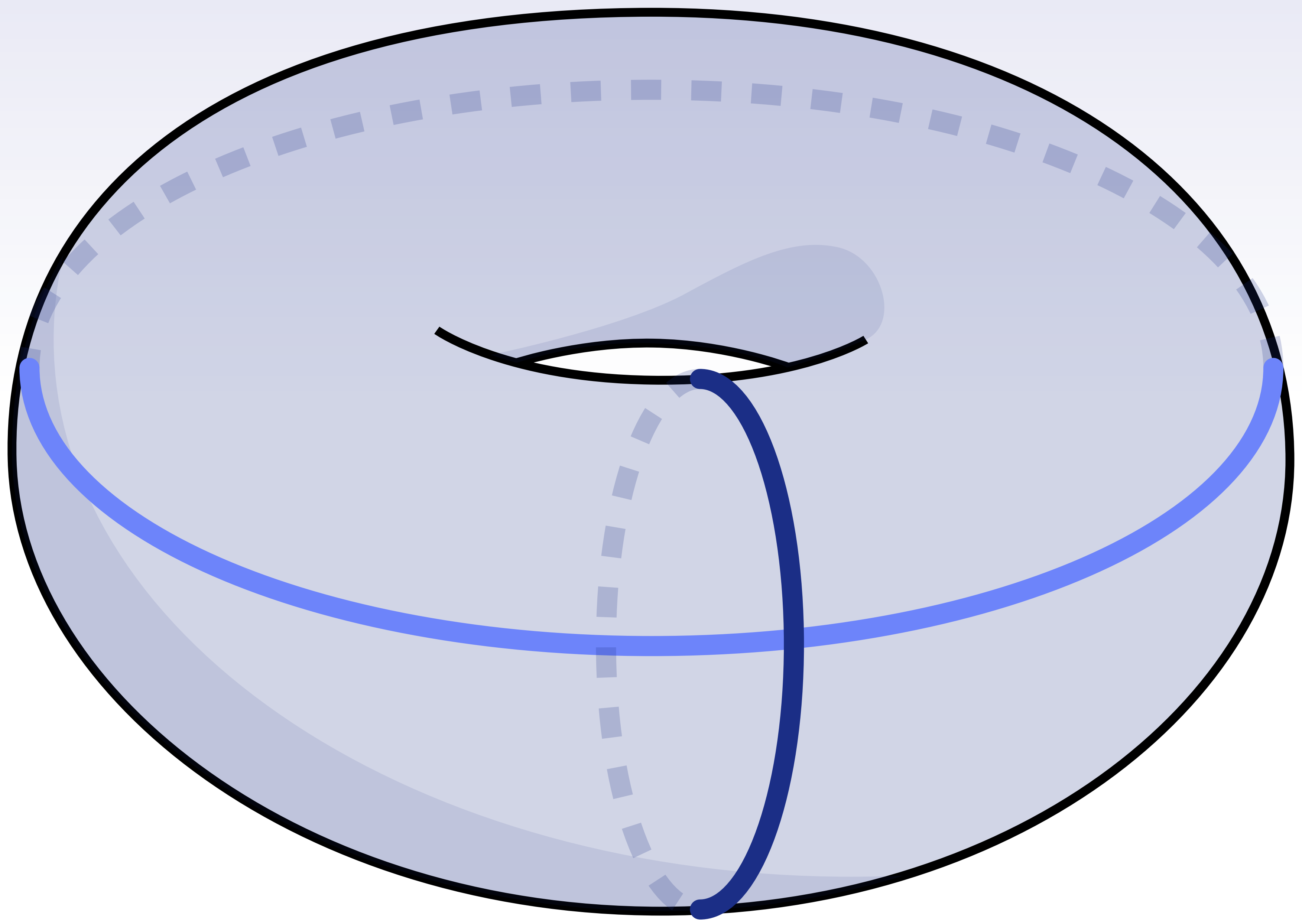


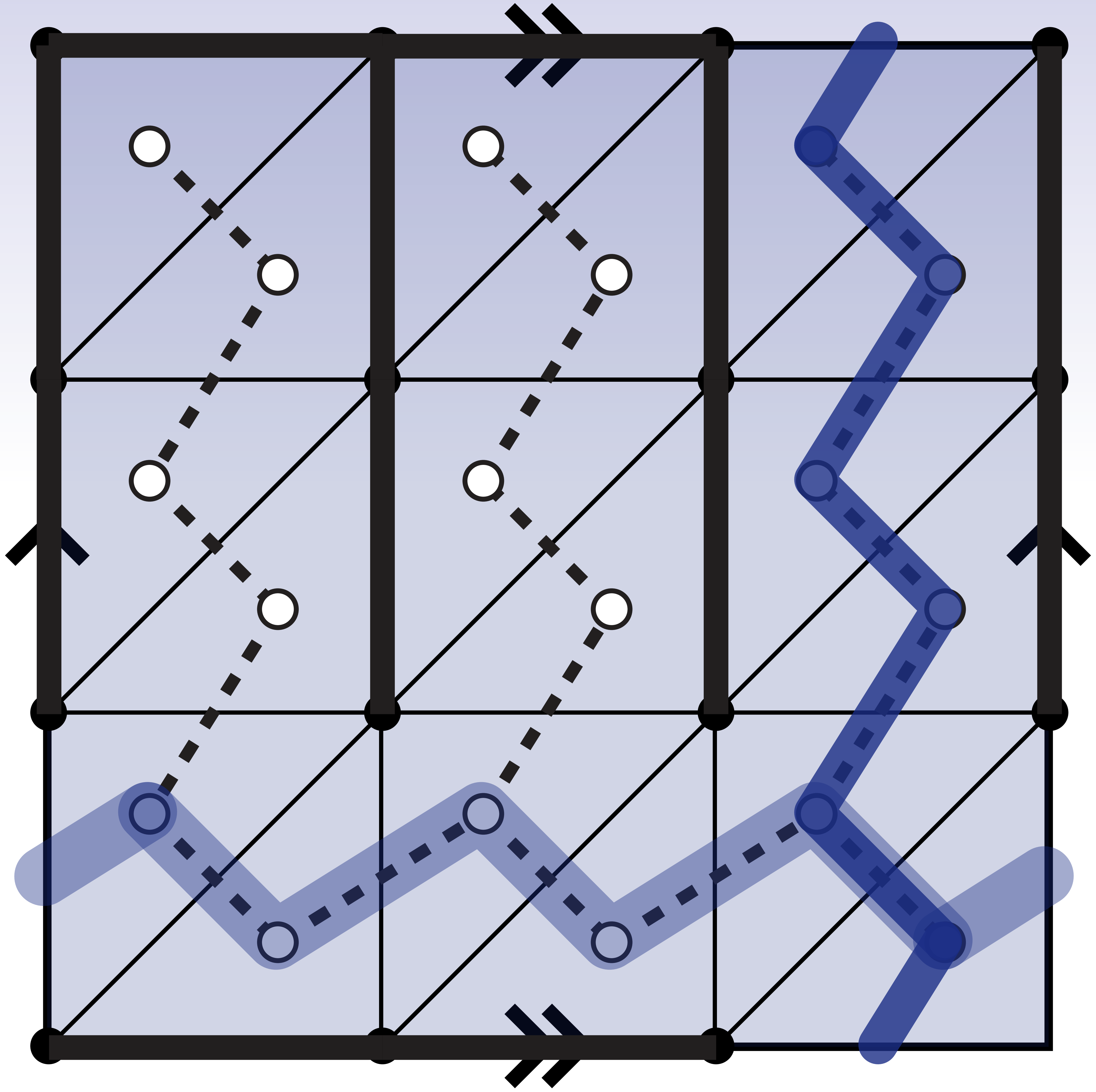
Example: Generators on Torus





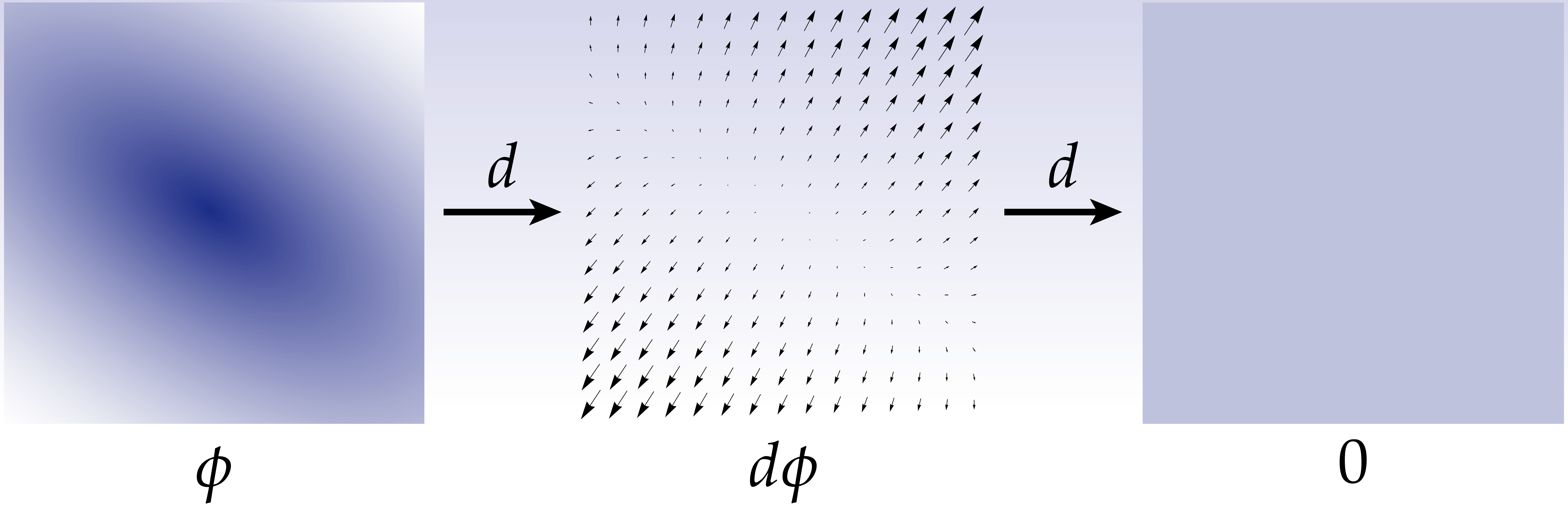
Example: Generators on Torus





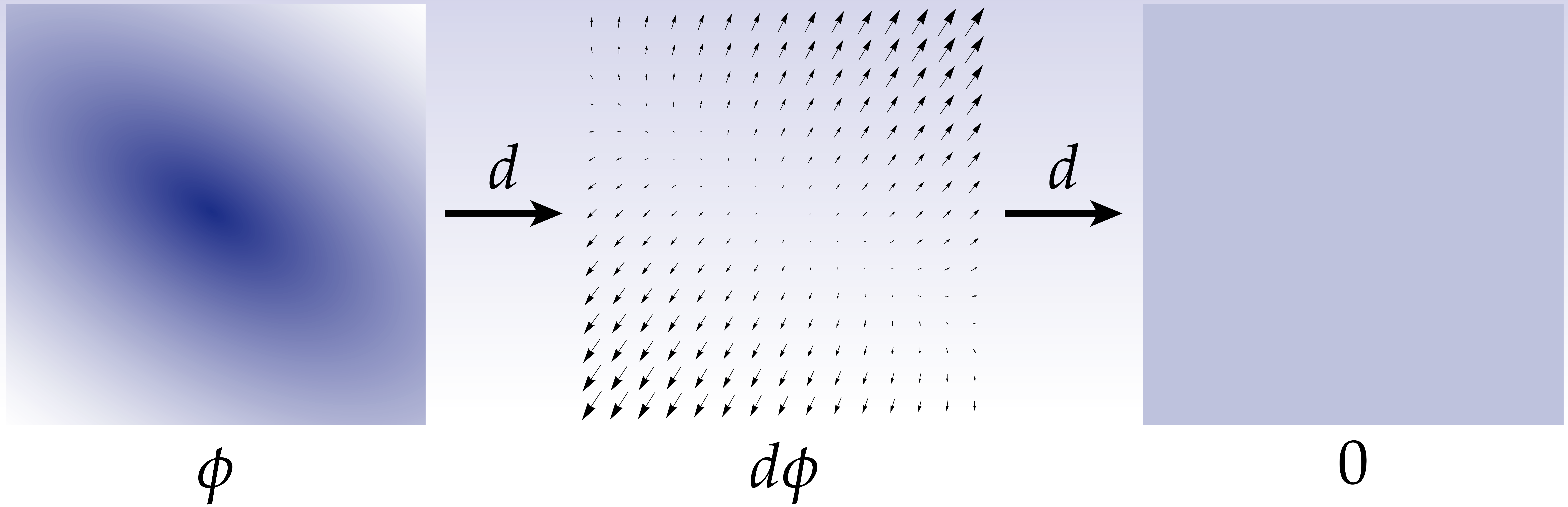
de Rham Cohomology

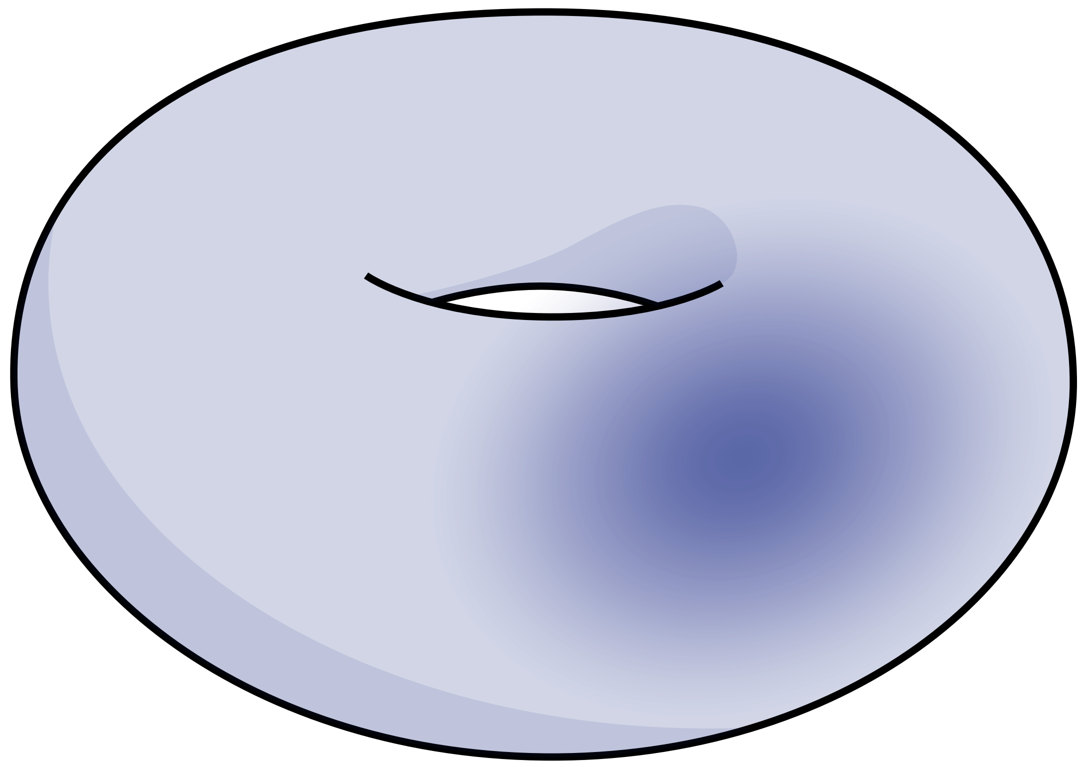
$d \circ d = 0$



de Rham Cohomology

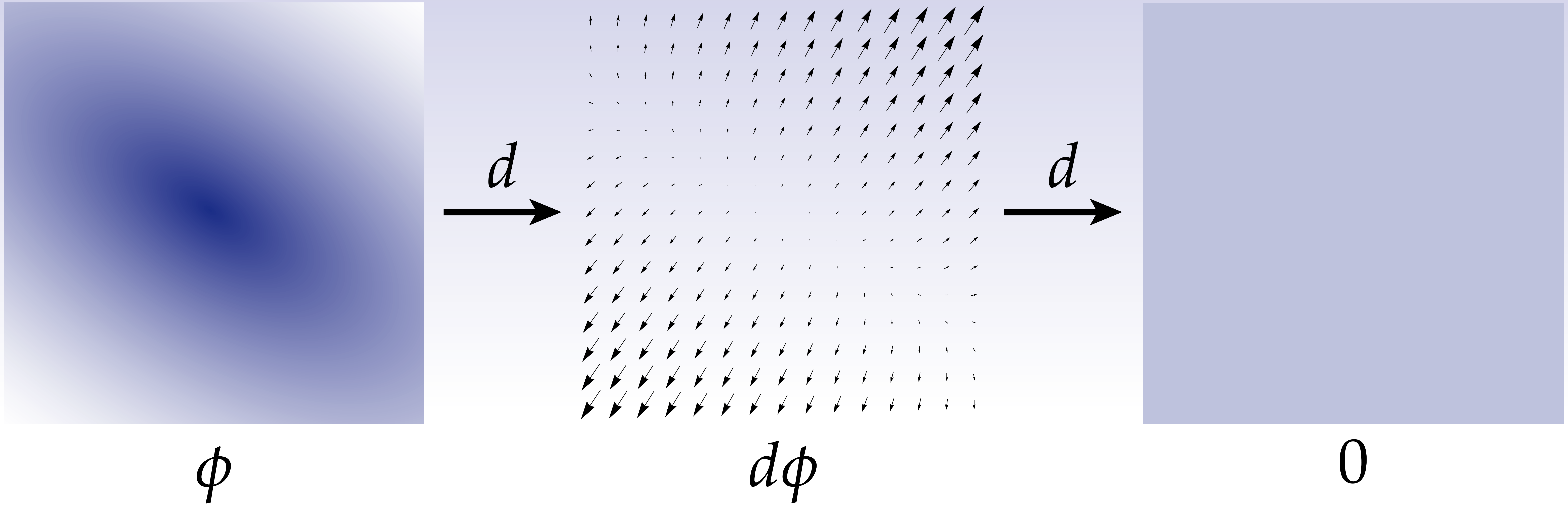
$$d \circ d = 0$$

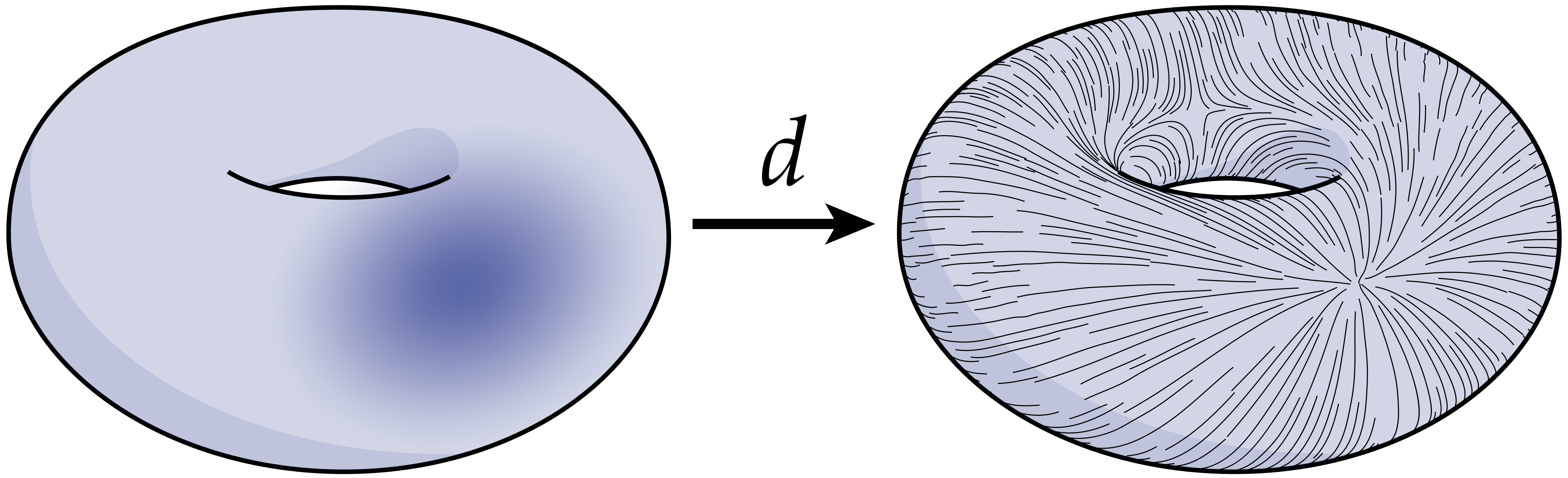




de Rham Cohomology

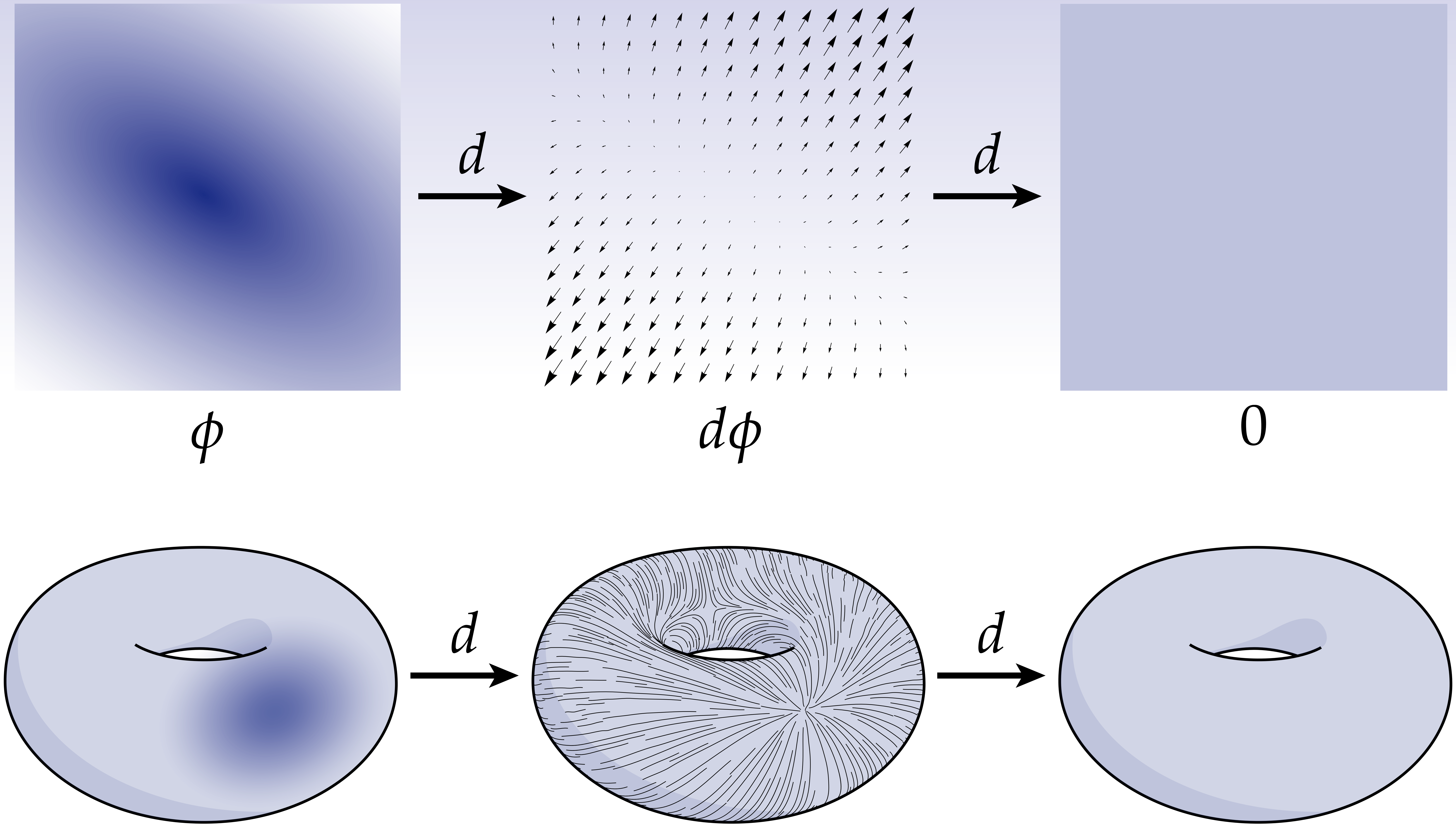
$$d \circ d = 0$$





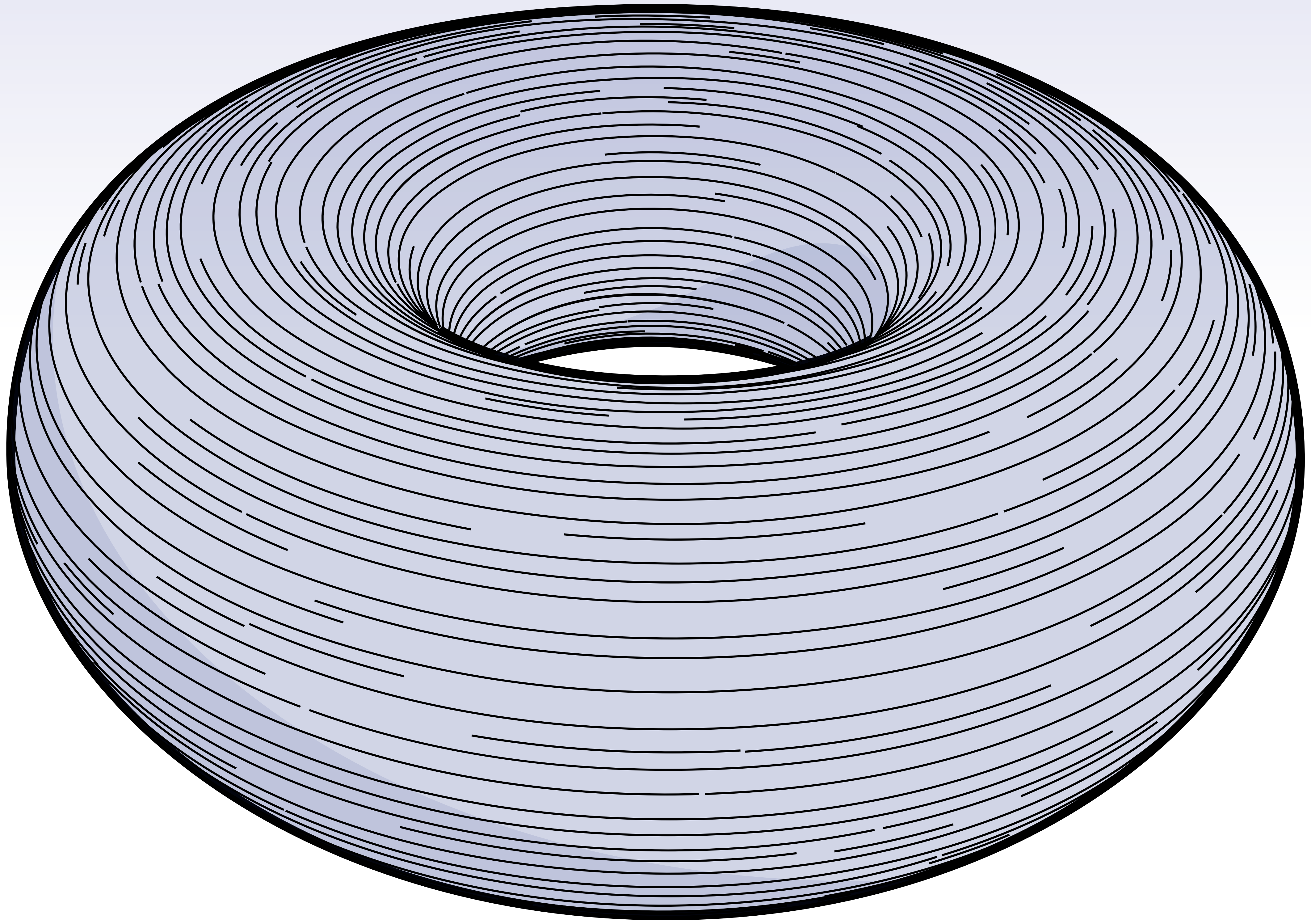
de Rham Cohomology

$$d \circ d = 0$$

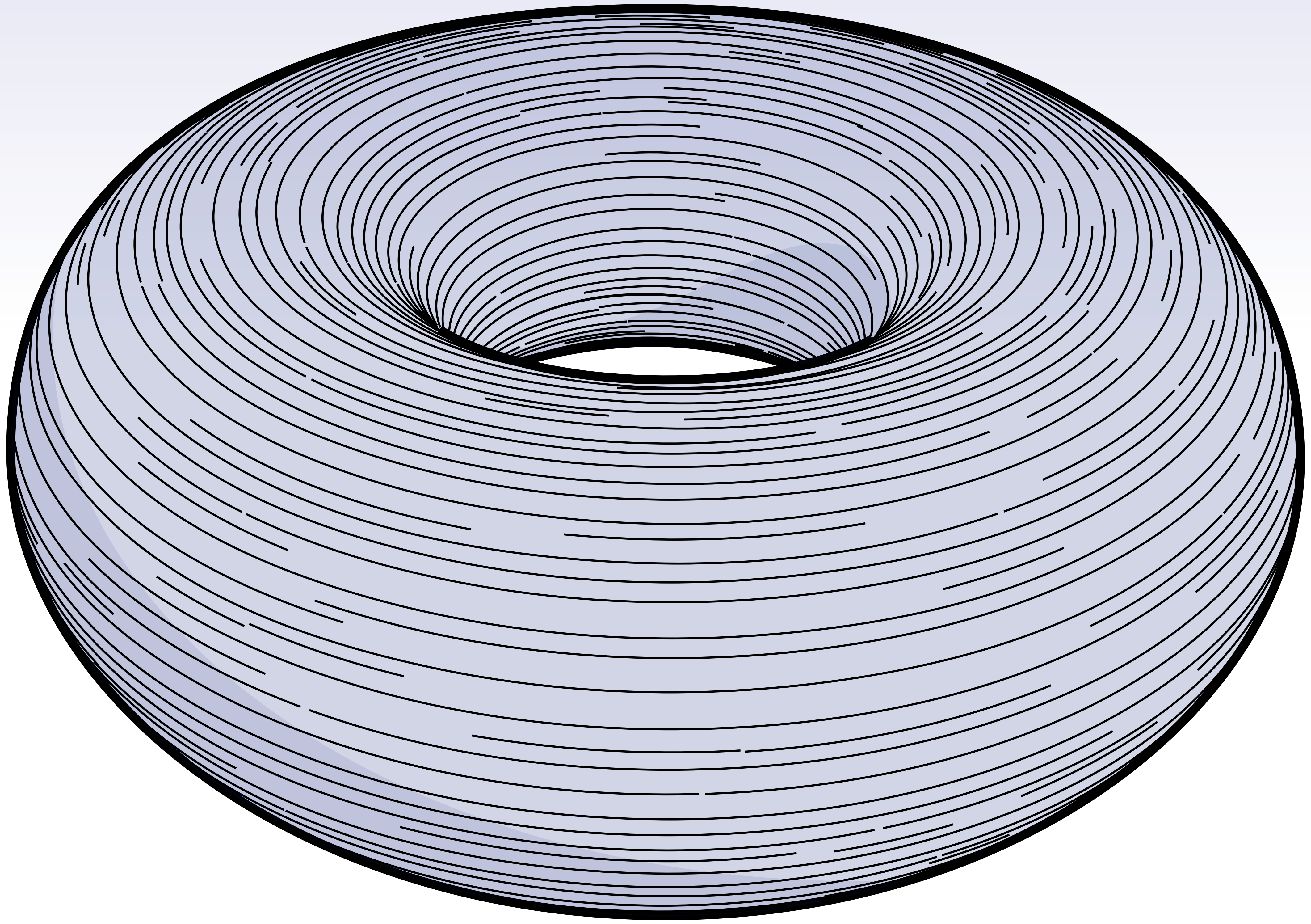


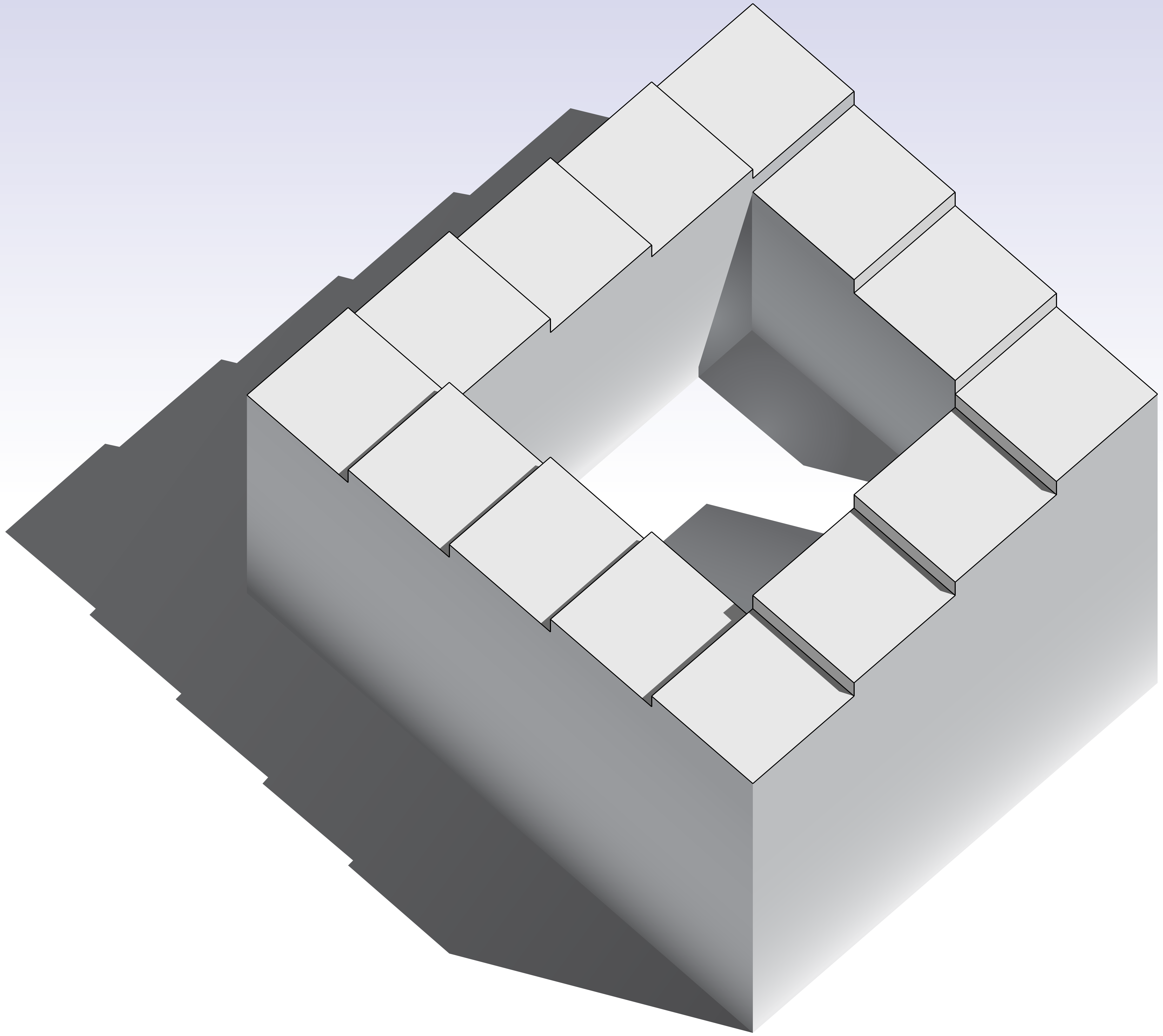
Harmonic 1-Forms

Harmonic 1-Forms

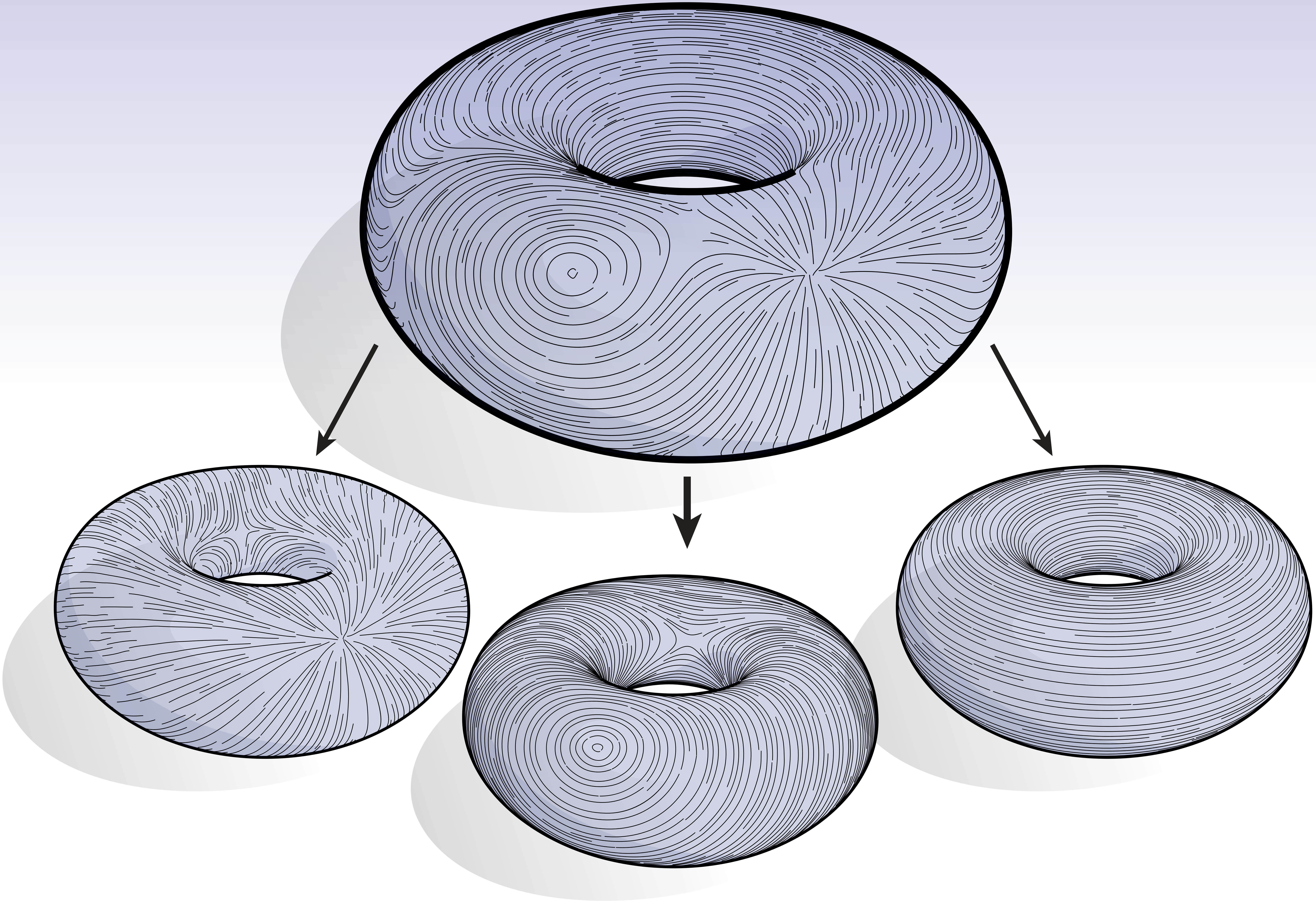


Harmonic 1-Forms

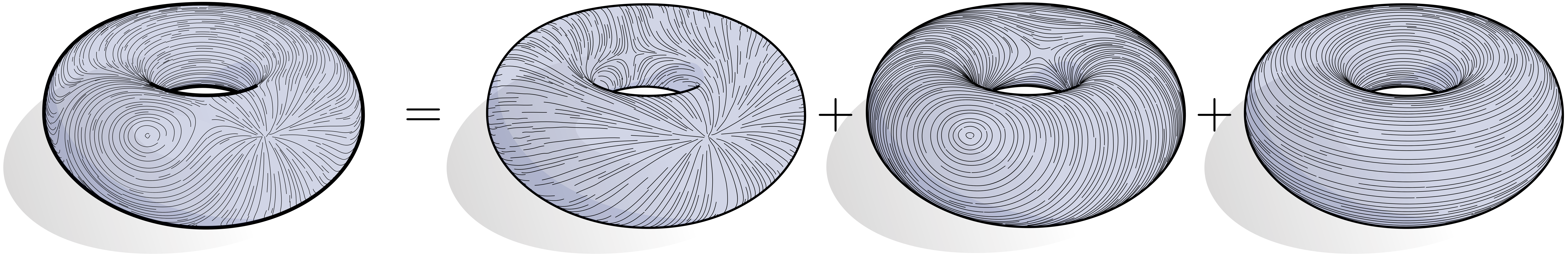




Helmholtz-Hodge Decomposition



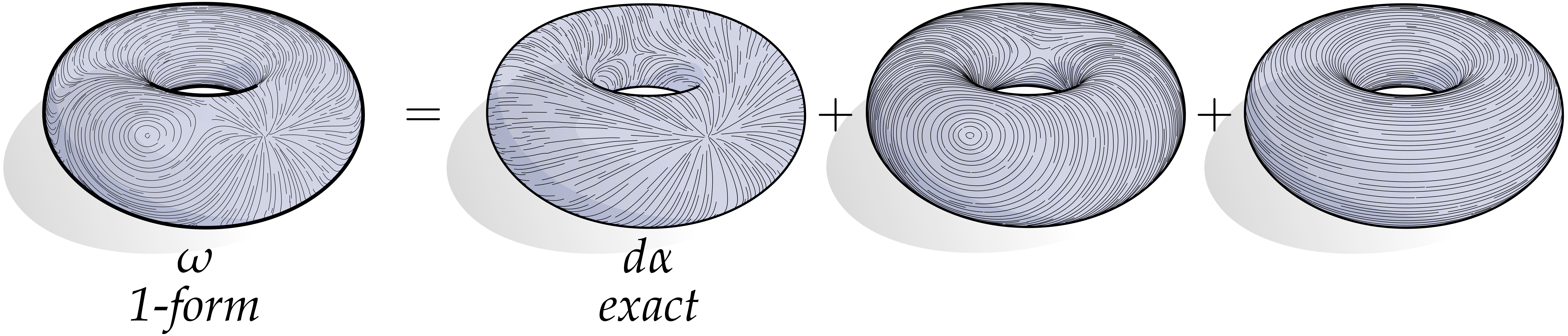
Helmholtz-Hodge Decomposition



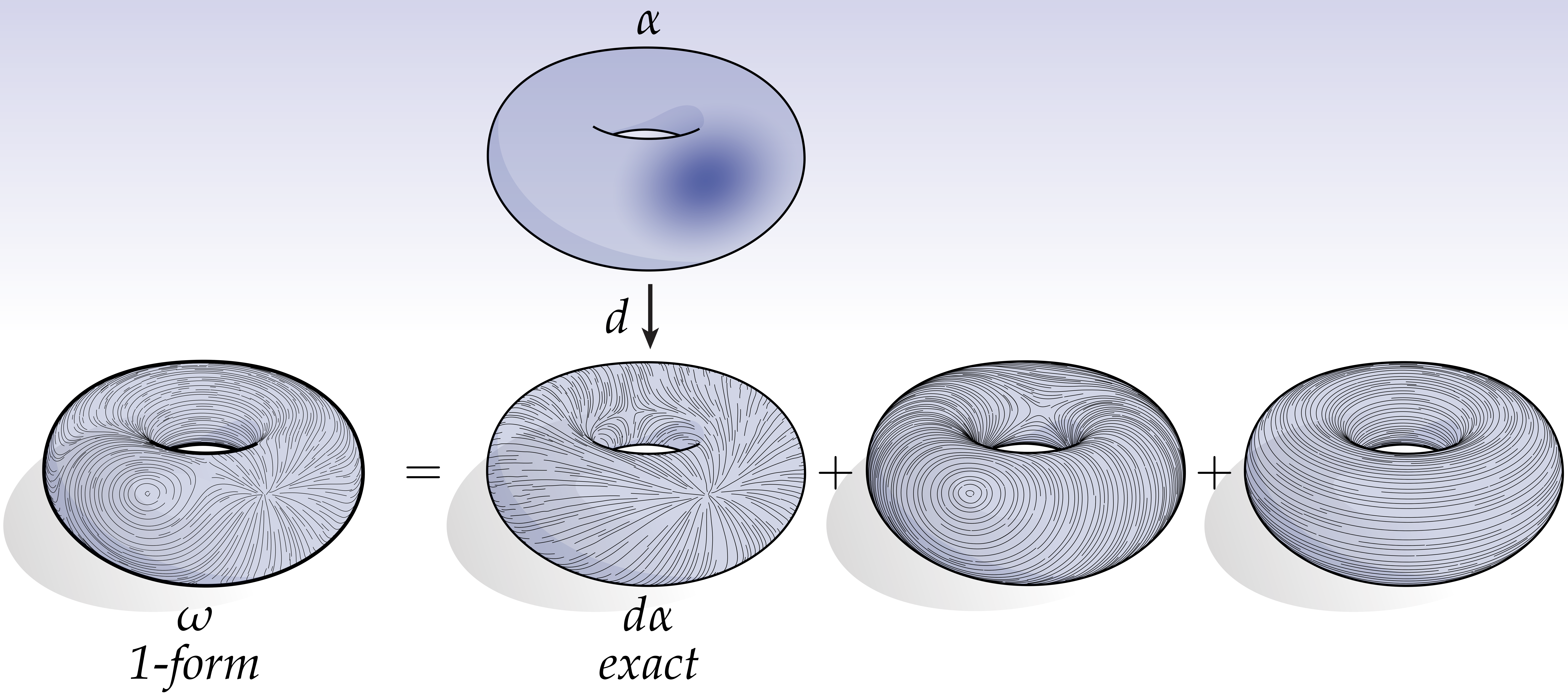
Helmholtz-Hodge Decomposition

ω
1-form

Helmholtz-Hodge Decomposition

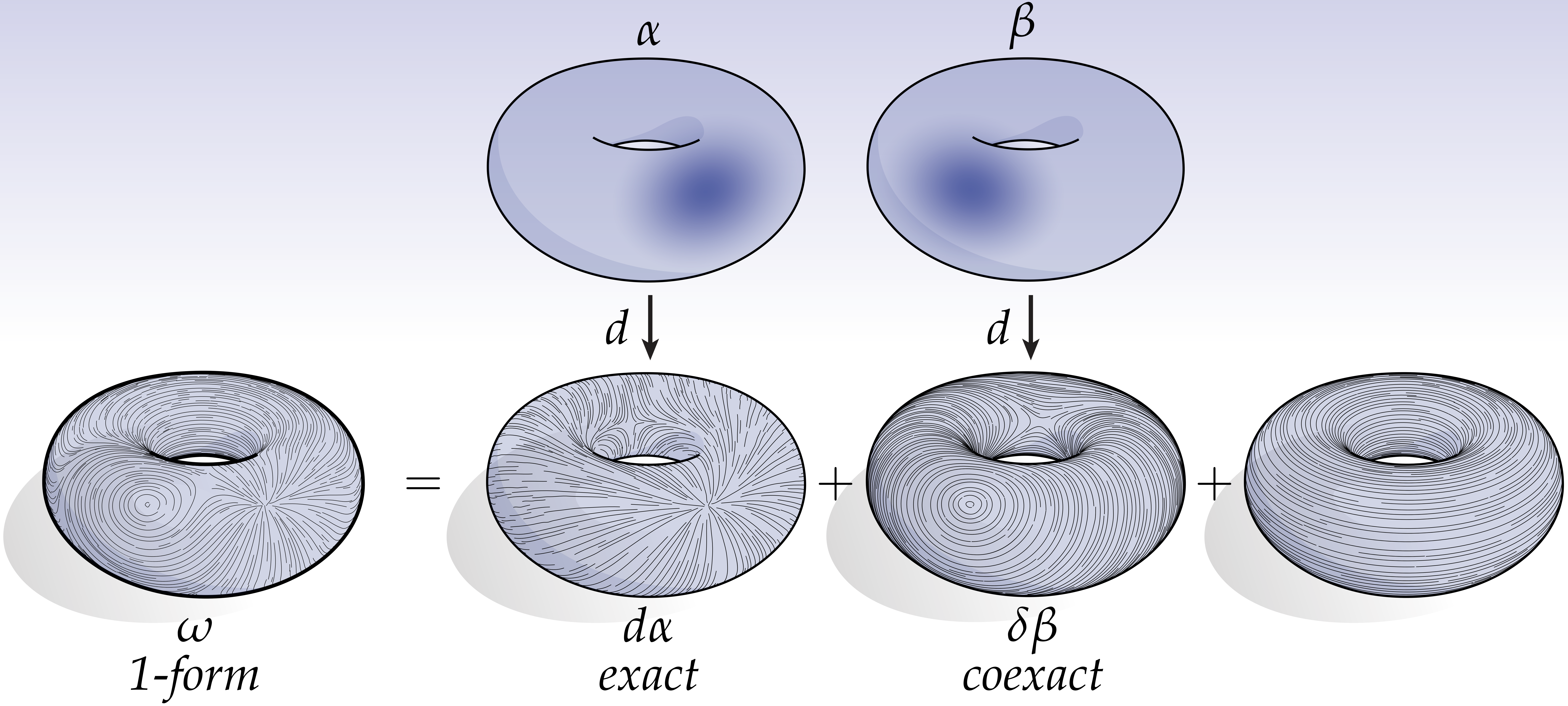


Helmholtz-Hodge Decomposition



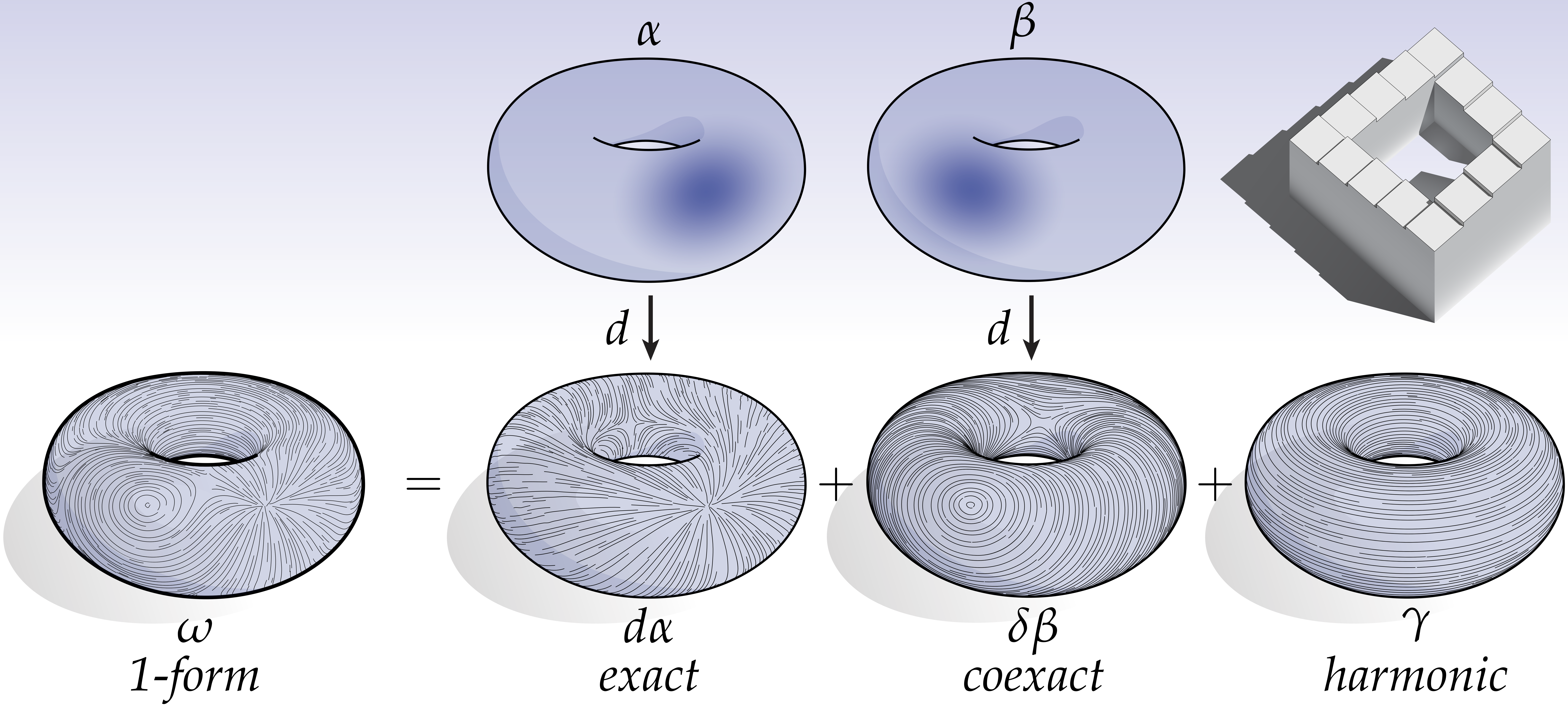
Helmholtz-Hodge Decomposition

Helmholtz-Hodge Decomposition

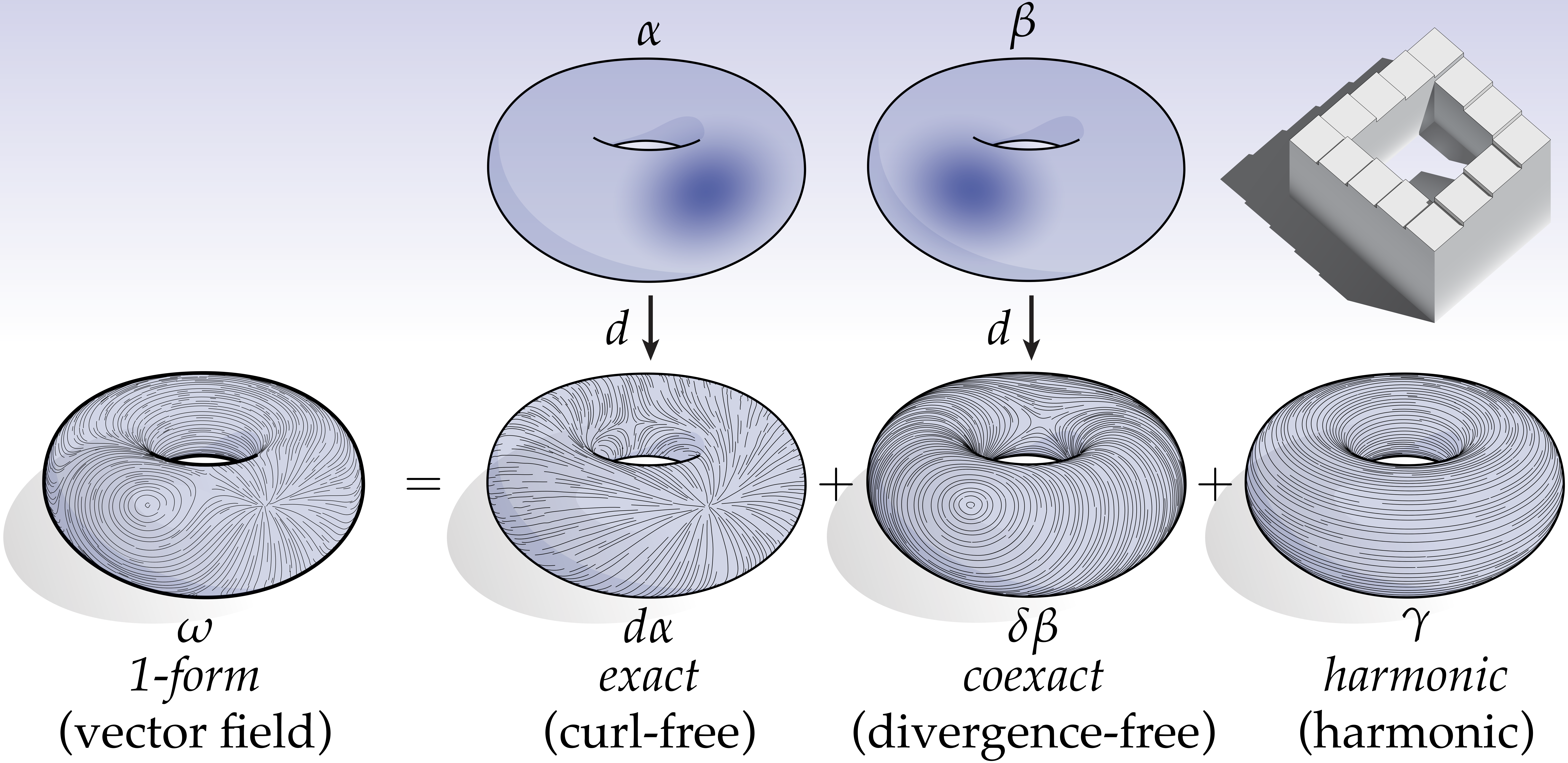


Helmholtz-Hodge Decomposition

Helmholtz-Hodge Decomposition



Helmholtz-Hodge Decomposition



Finding the Decomposition

$$\omega = d\alpha + \delta\beta + \gamma$$

Finding the Decomposition

$$d\omega = dd\alpha + d\delta\beta + d\gamma$$

Finding the Decomposition

$$d\omega = \cancelto{0}{dd}\alpha + d\delta\beta + d\gamma$$

Finding the Decomposition

$$d\omega = \cancelto{0}{dd}\alpha + d\delta\beta + \cancelto{0}{d}\gamma$$

Finding the Decomposition

$$d\omega = \qquad\qquad d\delta\beta$$

Finding the Decomposition

$$d\omega = d\delta\beta$$

Finding the Decomposition

$$d\omega = d\delta\beta$$
$$\delta\omega = \delta d\alpha$$

Finding the Decomposition

$$d\omega = d\delta\beta$$
$$\delta\omega = \Delta\alpha$$

Finding the Decomposition

$$d\omega = d\delta\beta$$

Poisson $\quad \Delta\alpha = \delta\omega$

Finding the Decomposition

$$d\omega = d\star d\star\beta$$

Poisson $$\Delta\alpha = \delta\omega$$

Finding the Decomposition

$$\star d\omega = \star d \star d \star \beta$$

Poisson $$\Delta \alpha = \delta \omega$$

Finding the Decomposition

$$\star d\omega = \star d \star d(\star\beta)$$

Poisson $$\Delta\alpha = \delta\omega$$

Finding the Decomposition

$$\star d\omega = \star d \star d\tilde{\beta}$$

Poisson $$\Delta\alpha = \delta\omega$$

Finding the Decomposition

$$\star d\omega = \Delta\tilde{\beta}$$

Poisson $$\Delta\alpha = \delta\omega$$

Finding the Decomposition

Poisson $$\Delta\tilde{\beta} = \star d\omega$$

Poisson $$\Delta\alpha = \delta\omega$$

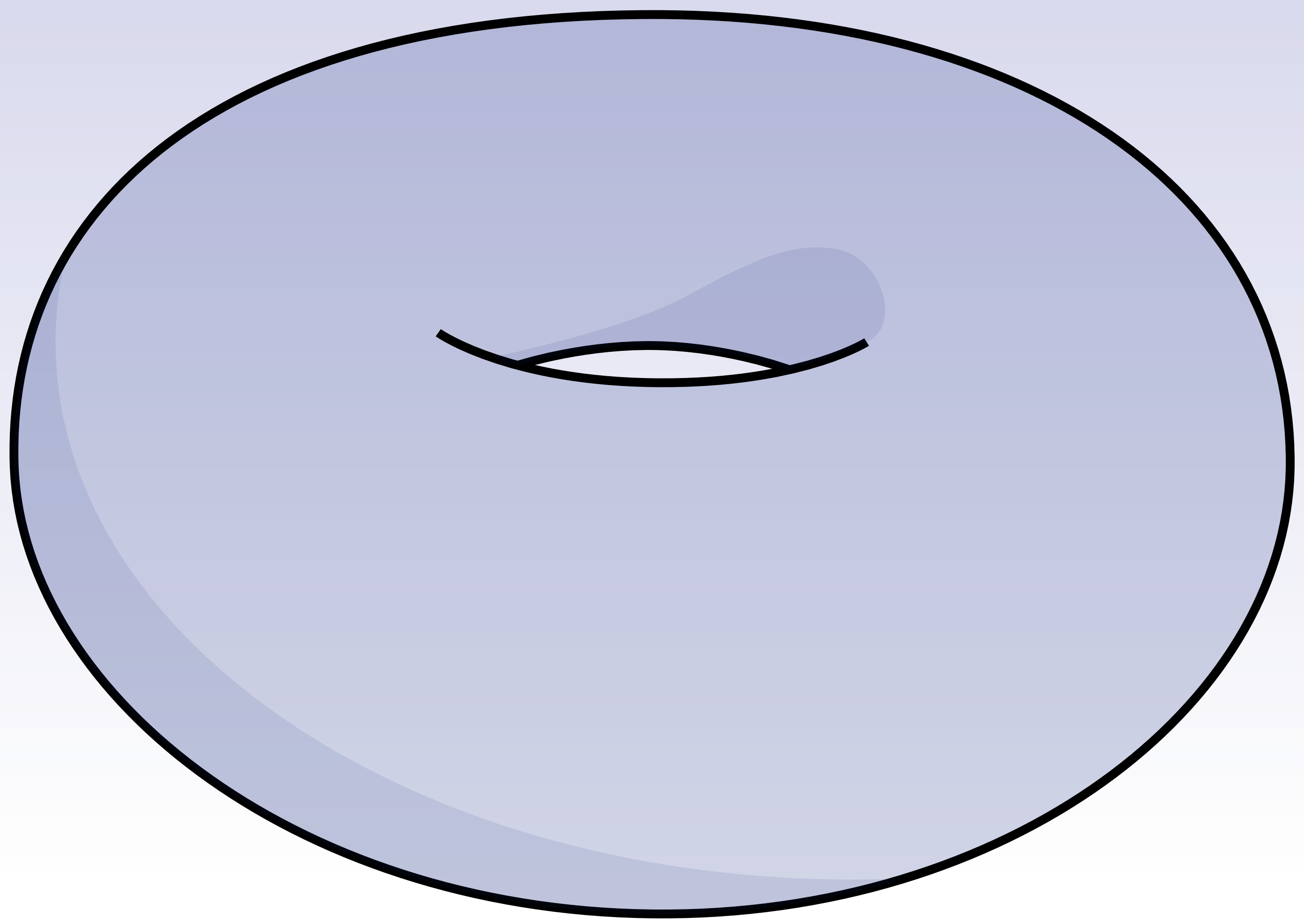
Finding the Decomposition

Poisson $\Delta\tilde{\beta} = \star d\omega \qquad (\beta = \star\tilde{\beta})$

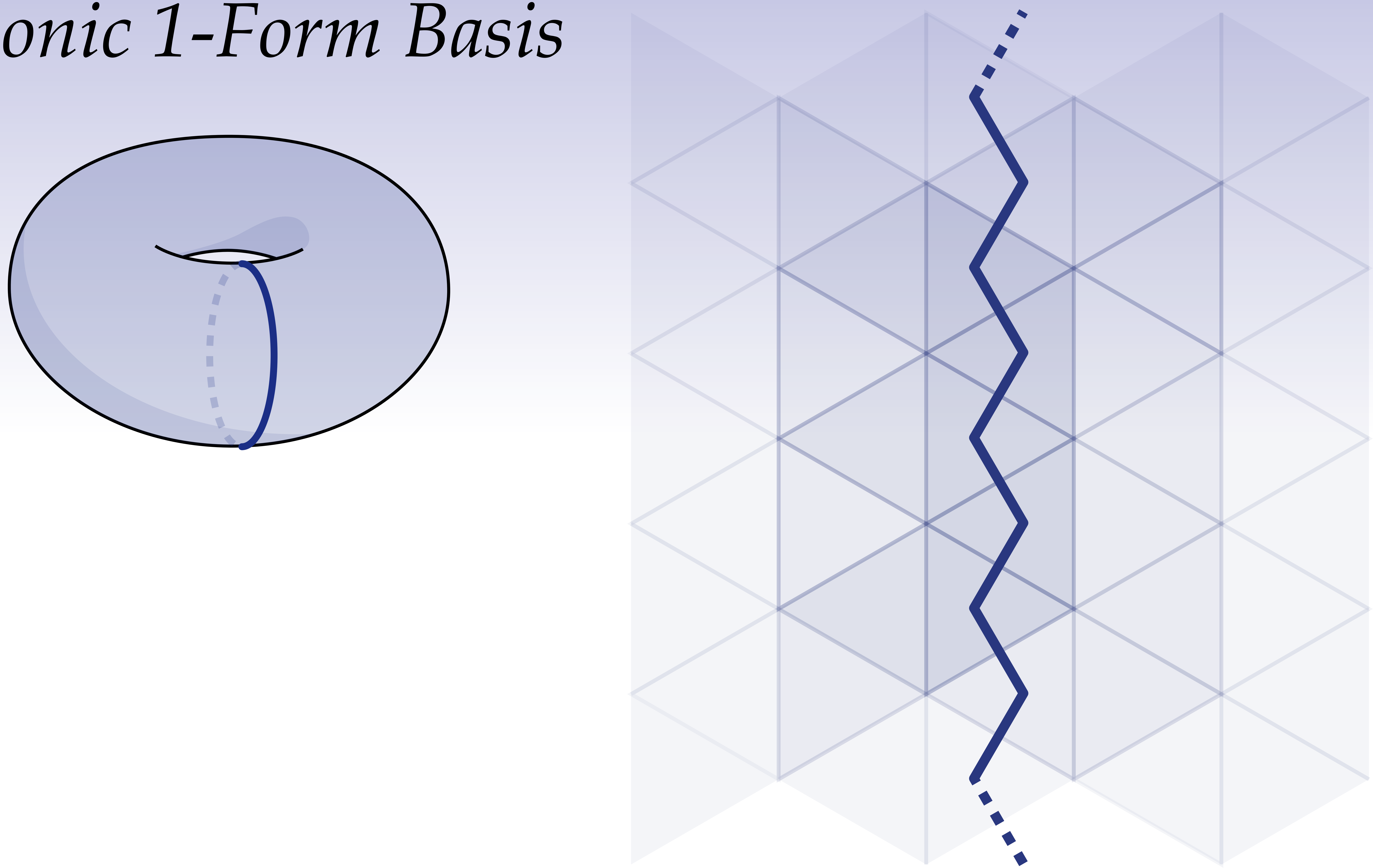
Poisson $\Delta\alpha = \delta\omega$

Harmonic 1-Form Basis

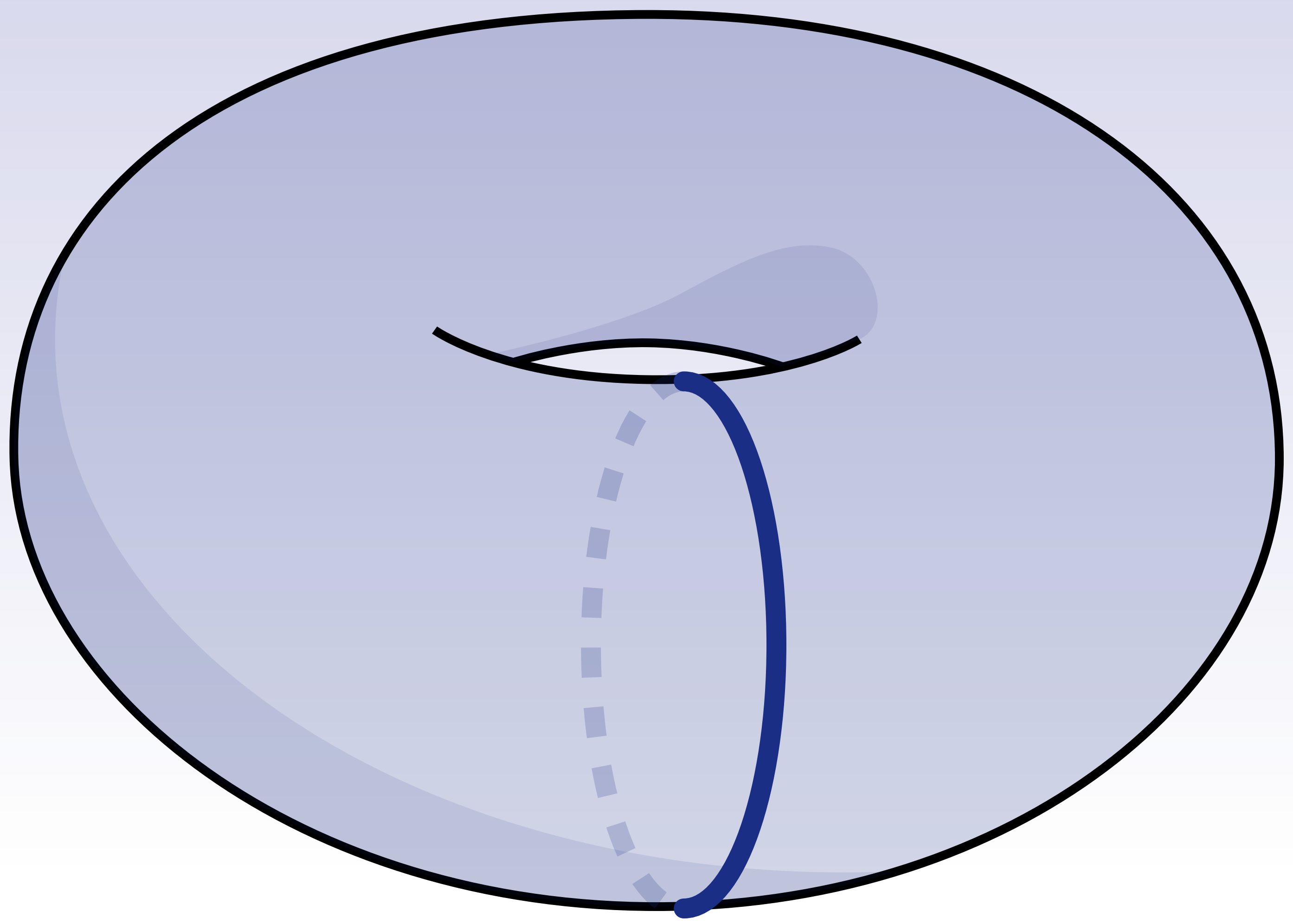
Harmonic 1-Form Basis

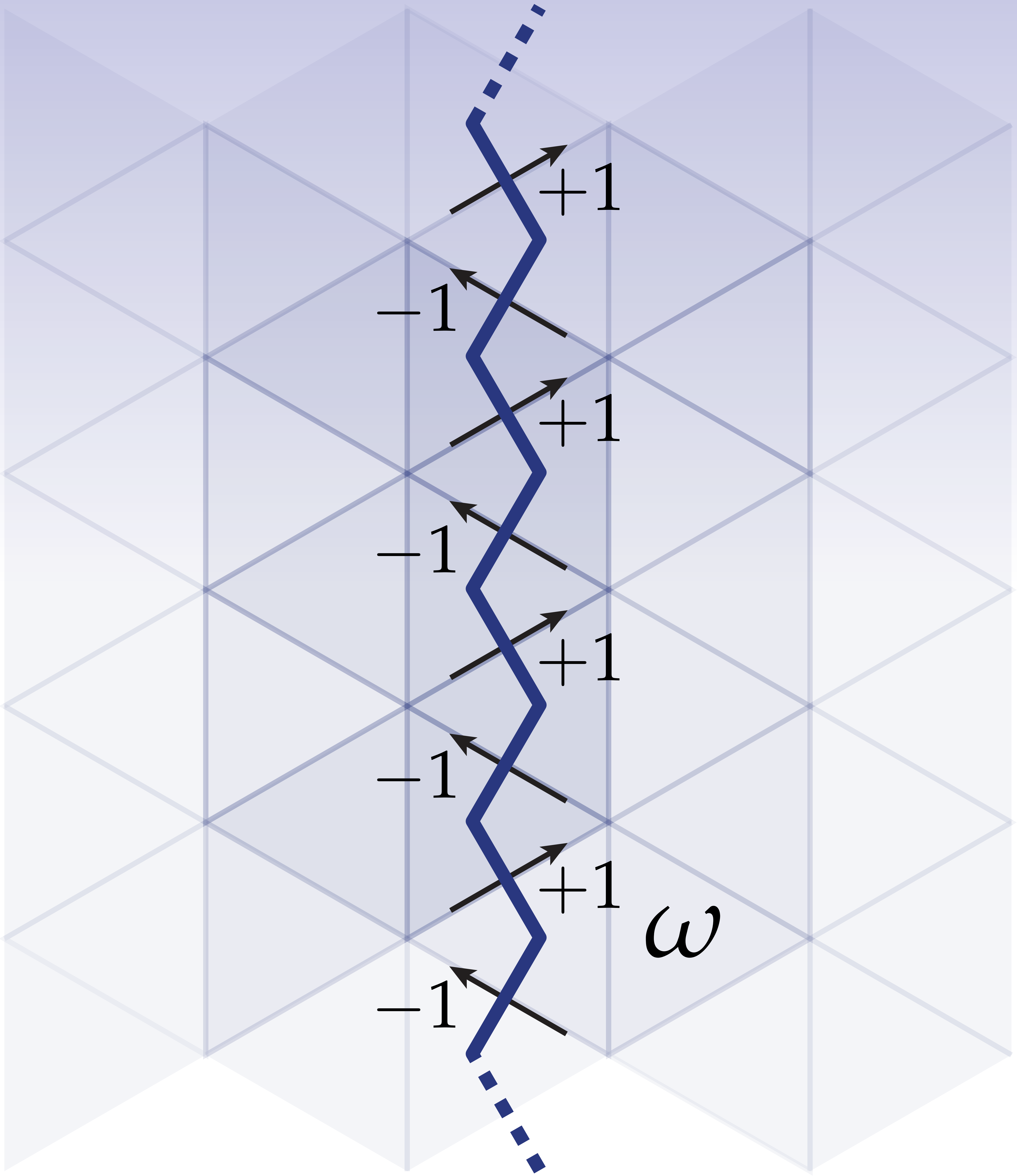


Harmonic 1-Form Basis

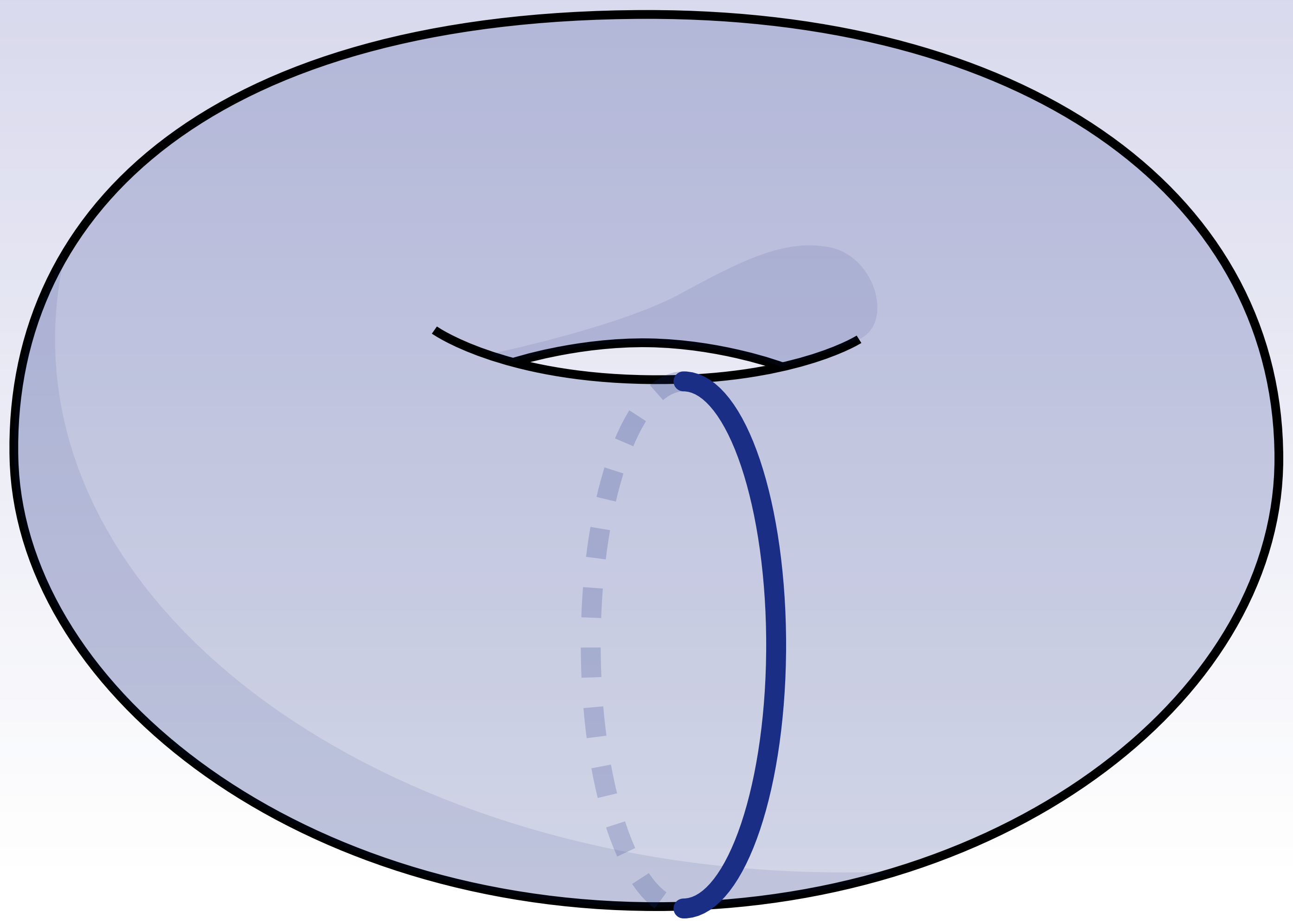


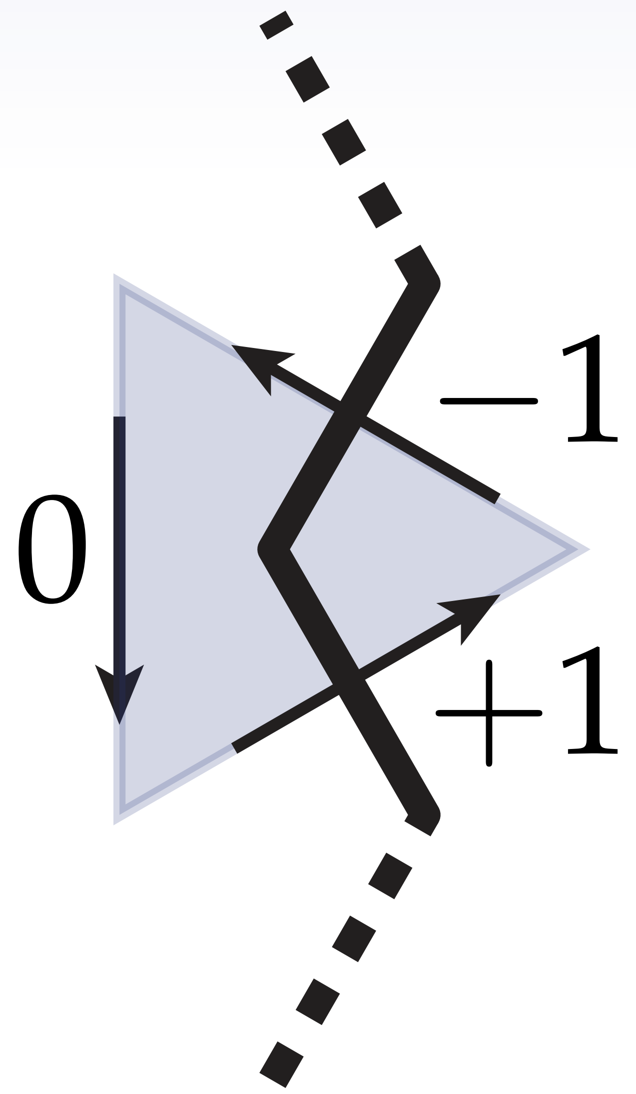
Harmonic 1-Form Basis



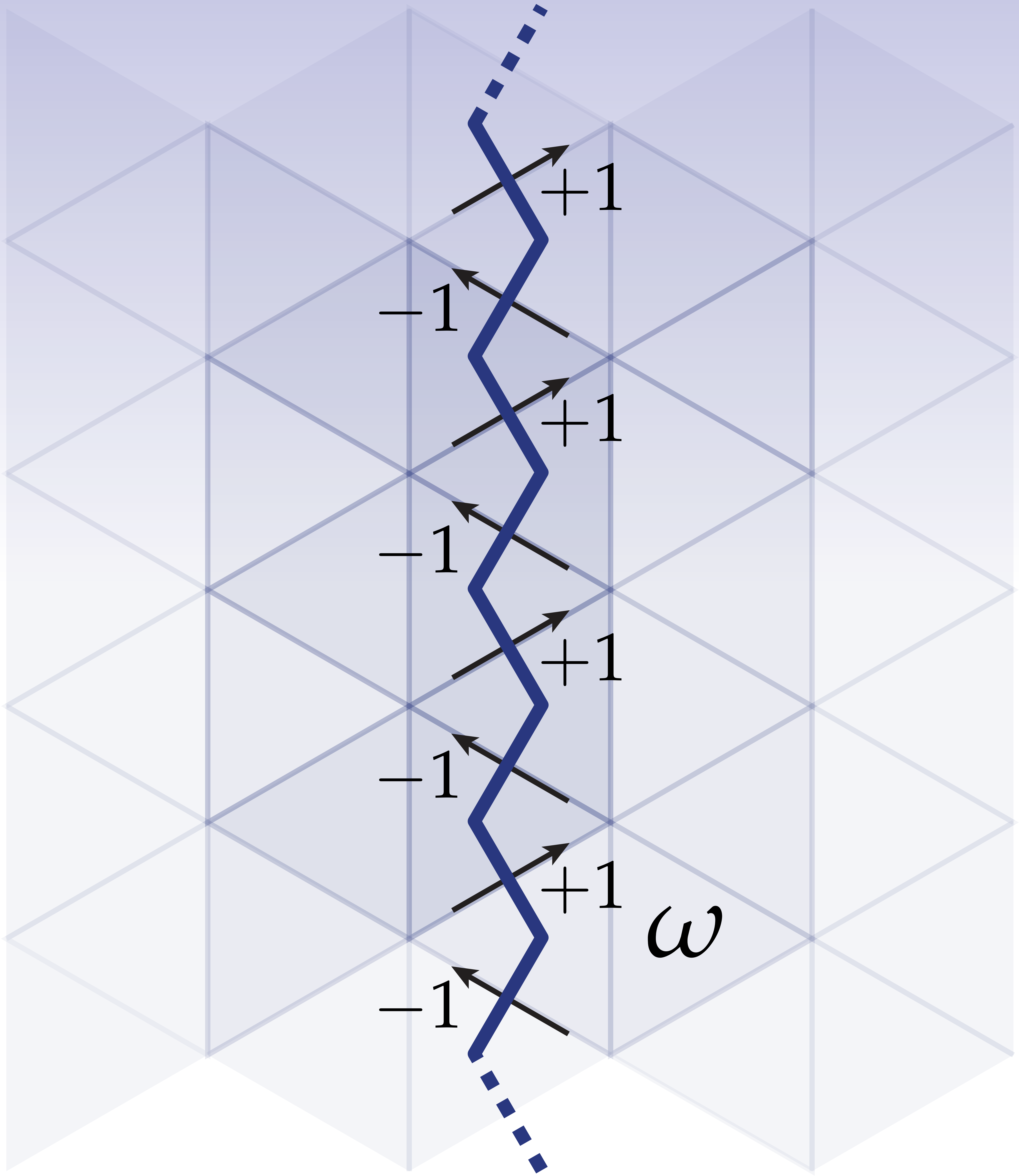


Harmonic 1-Form Basis

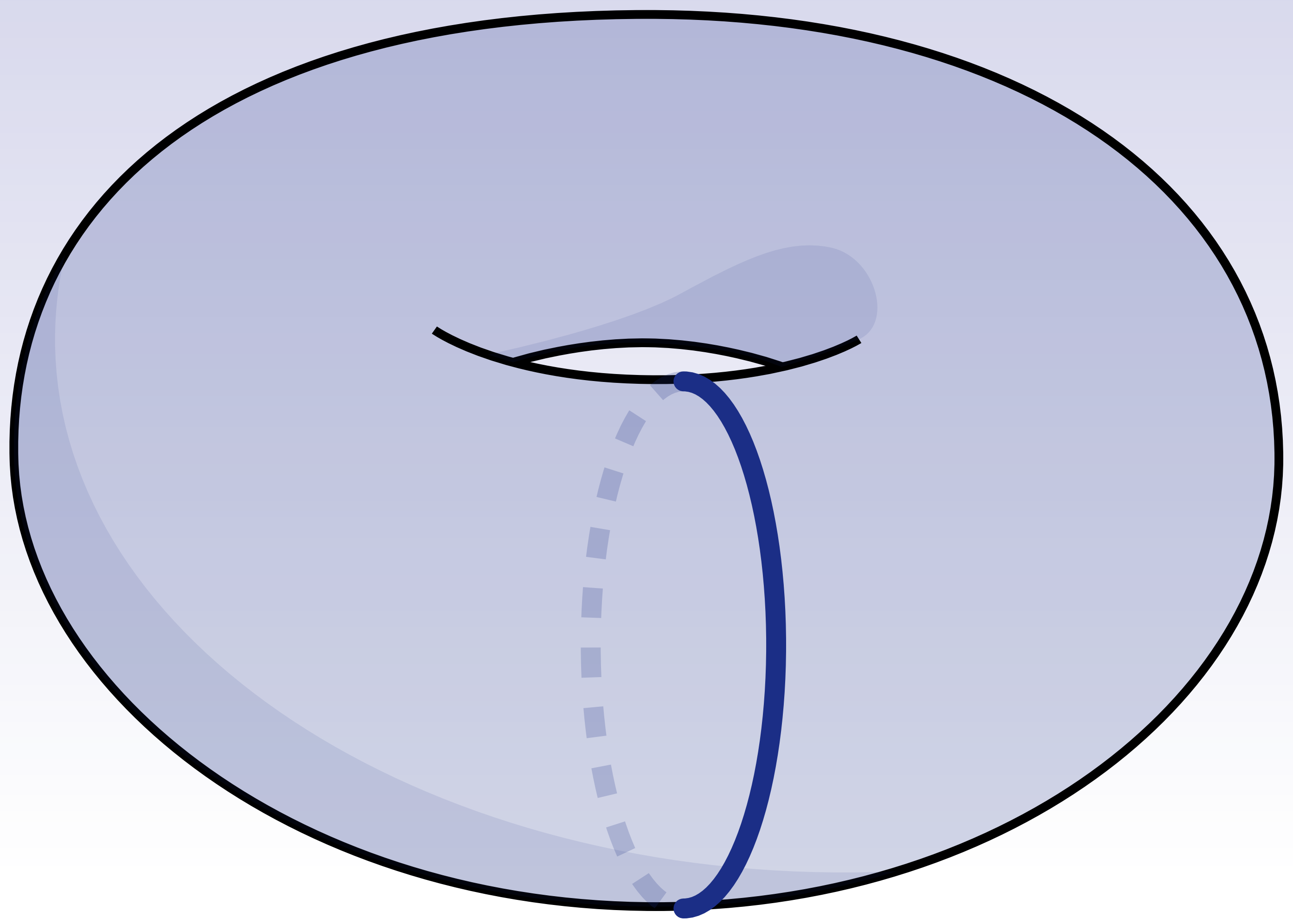


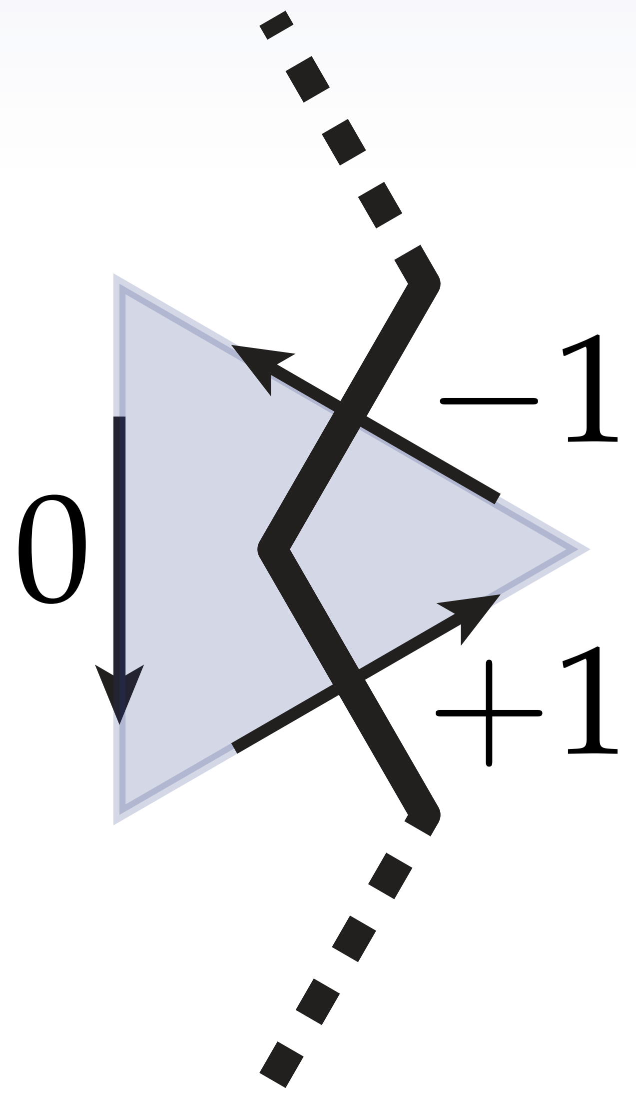


$$d\omega = 0$$



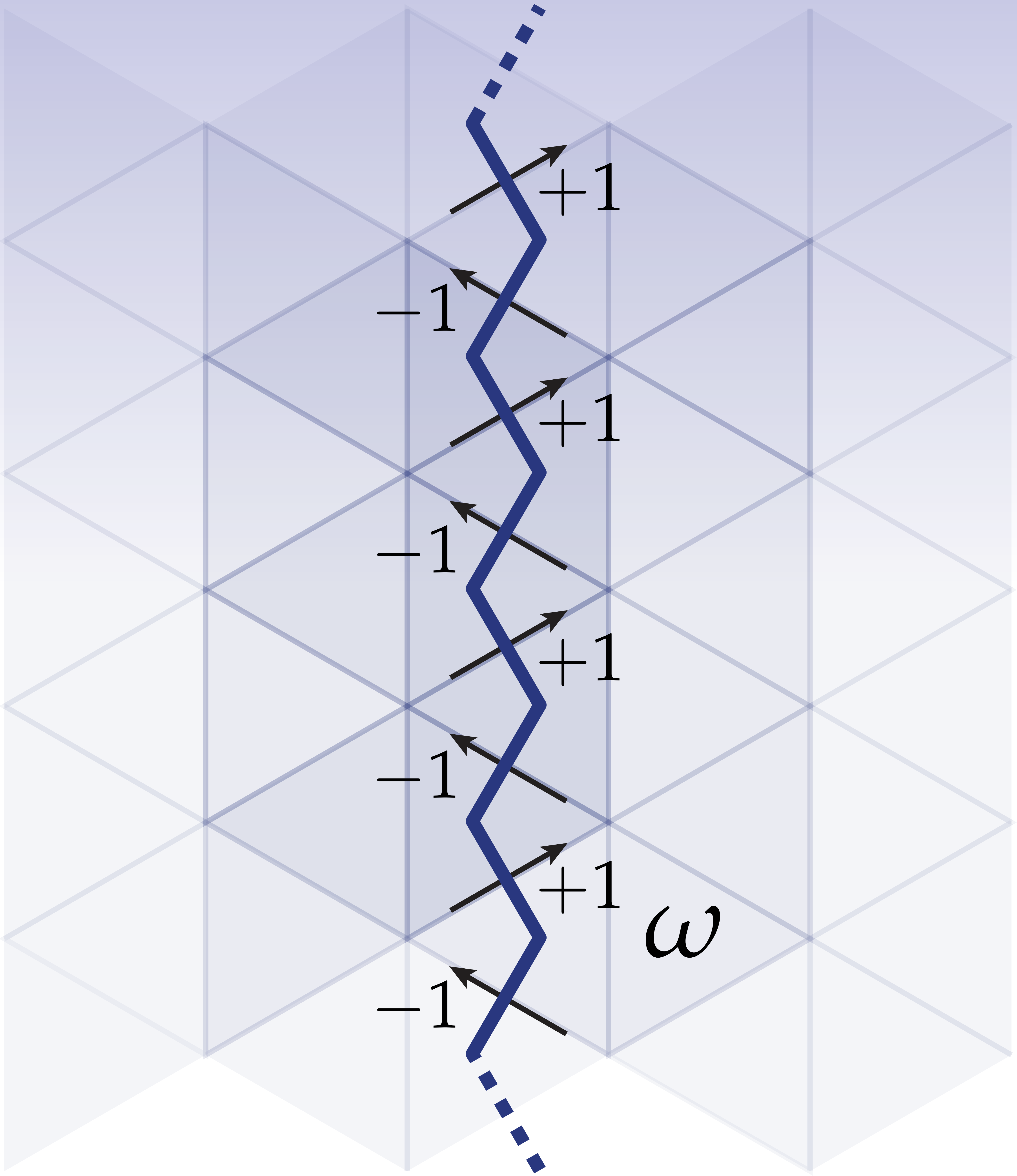
Harmonic 1-Form Basis



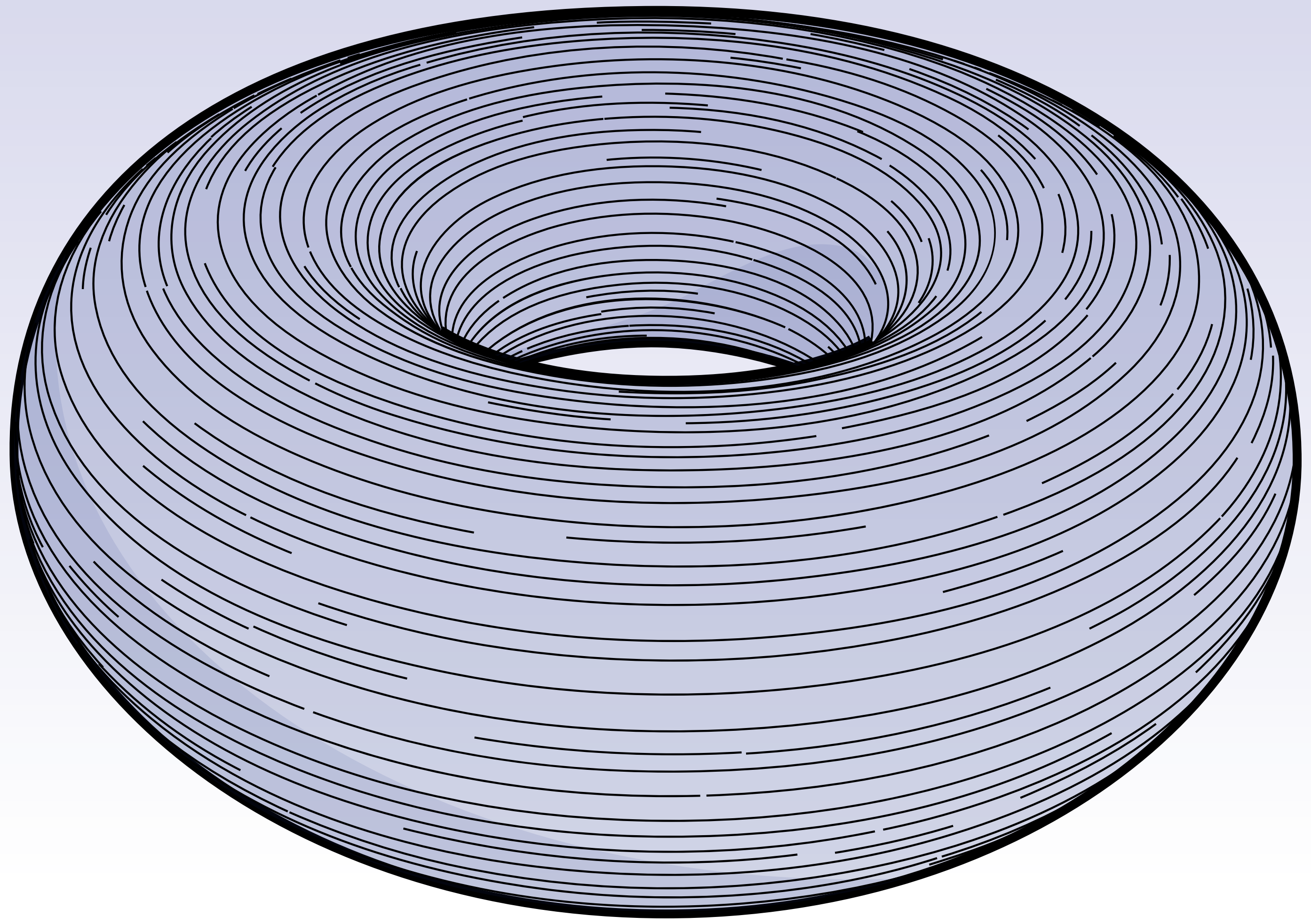


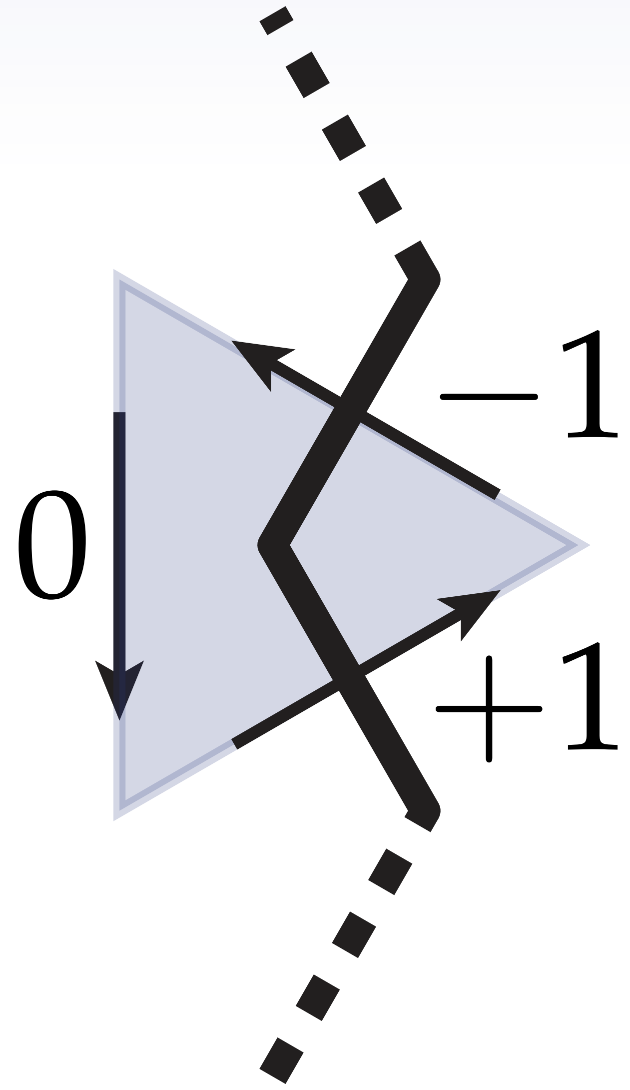
$$d\omega = 0$$

$$\Rightarrow \omega = d\alpha + \gamma$$



Harmonic 1-Form Basis

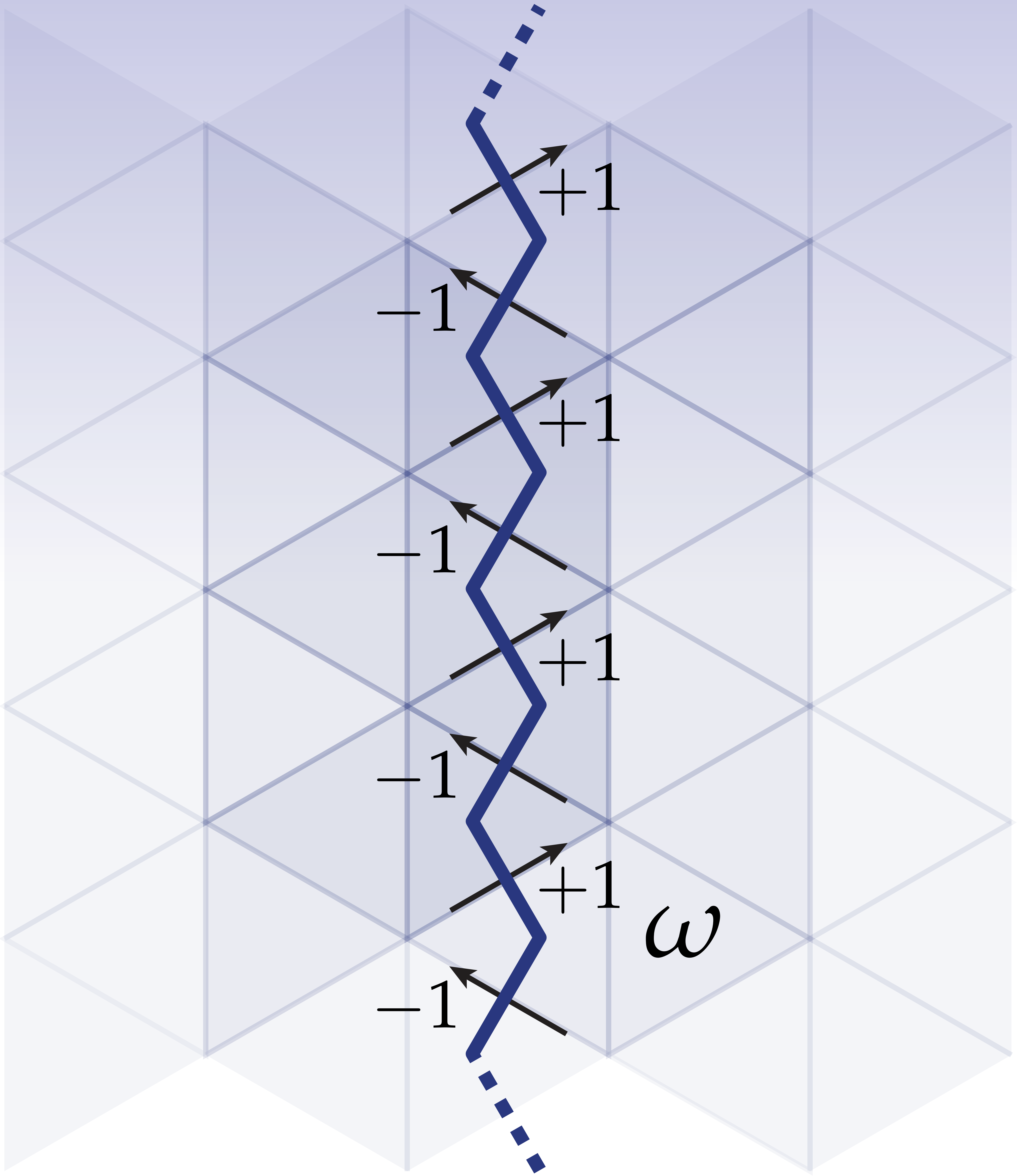




$$d\omega = 0$$

$$\Rightarrow \omega = d\alpha + \gamma$$

$$\boxed{\gamma \leftarrow \omega - d\alpha}$$



Homology & Cohomology - Summary

Homology & Cohomology - Summary

- Detect “interesting part” of a space

Homology & Cohomology - Summary

- Detect “interesting part” of a space
  - regions & loops (*singular homology*)

Homology & Cohomology - Summary

- Detect "interesting part" of a space
  - regions & loops (*singular homology*)
  - differential forms (*de Rham cohomology*)

Homology & Cohomology - Summary

- Detect "interesting part" of a space
  - regions & loops (*singular homology*)
  - differential forms (*de Rham cohomology*)
- Two simple algorithms

Homology & Cohomology - Summary

- Detect "interesting part" of a space
  - regions & loops (*singular homology*)
  - differential forms (*de Rham cohomology*)
- Two simple algorithms
  - loops: *tree-cotree decomposition*

Homology & Cohomology - Summary

- Detect “interesting part” of a space
  - regions & loops (*singular homology*)
  - differential forms (*de Rham cohomology*)
- Two simple algorithms
  - loops: *tree-cotree decomposition*
  - forms: *Helmholtz-Hodge decomposition*

Homology & Cohomology - Summary

- Detect “interesting part” of a space
  - regions & loops (*singular homology*)
  - differential forms (*de Rham cohomology*)
- Two simple algorithms
  - loops: *tree-cotree decomposition*
  - forms: *Helmholtz-Hodge decomposition*
    - more generally, any 1-form

Homology & Cohomology - Summary

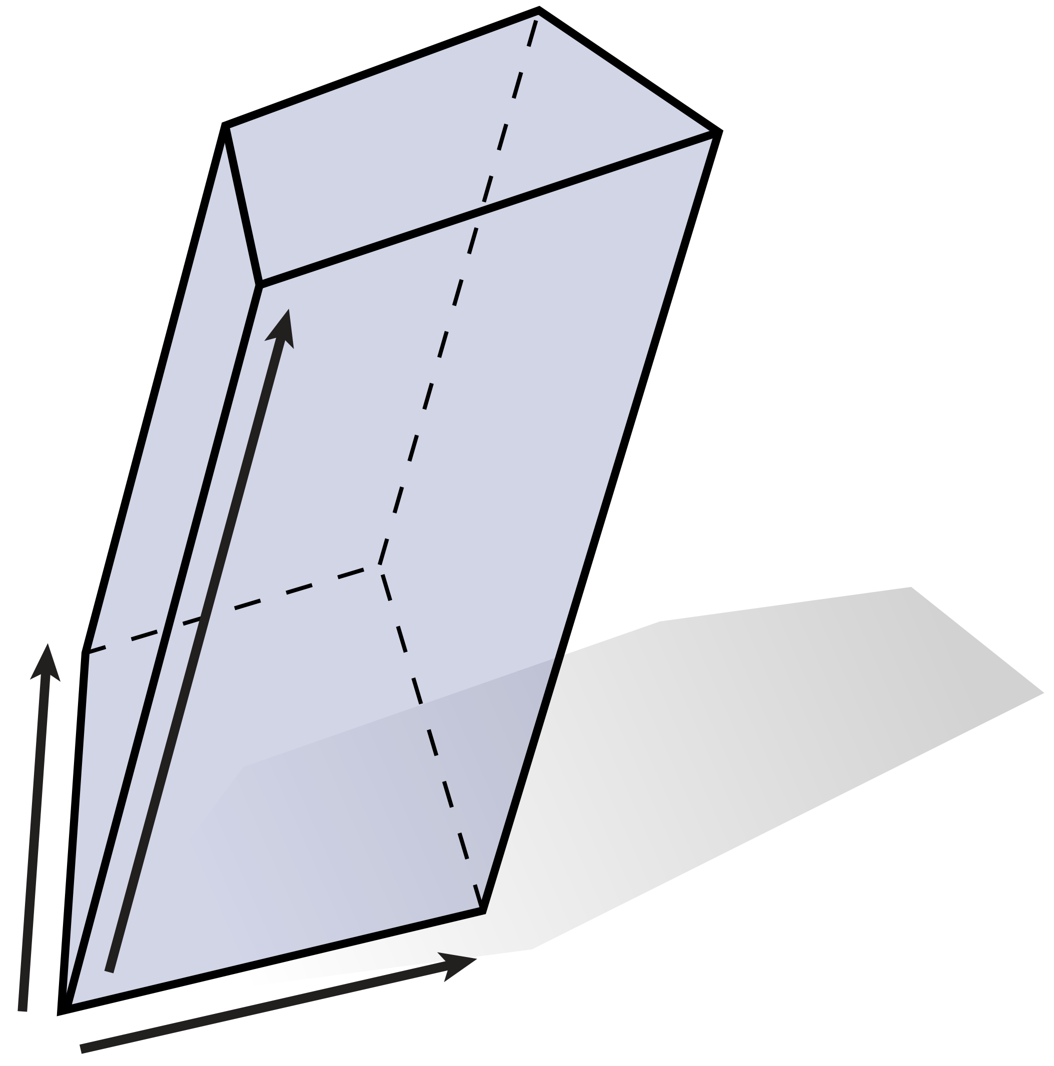
- Detect "interesting part" of a space
  - regions & loops (*singular homology*)
  - differential forms (*de Rham cohomology*)
- Two simple algorithms
  - loops: *tree-cotree decomposition*
  - forms: *Helmholtz-Hodge decomposition*
    - more generally, any 1-form
    - boils down to *Poisson equation*

Conclusion

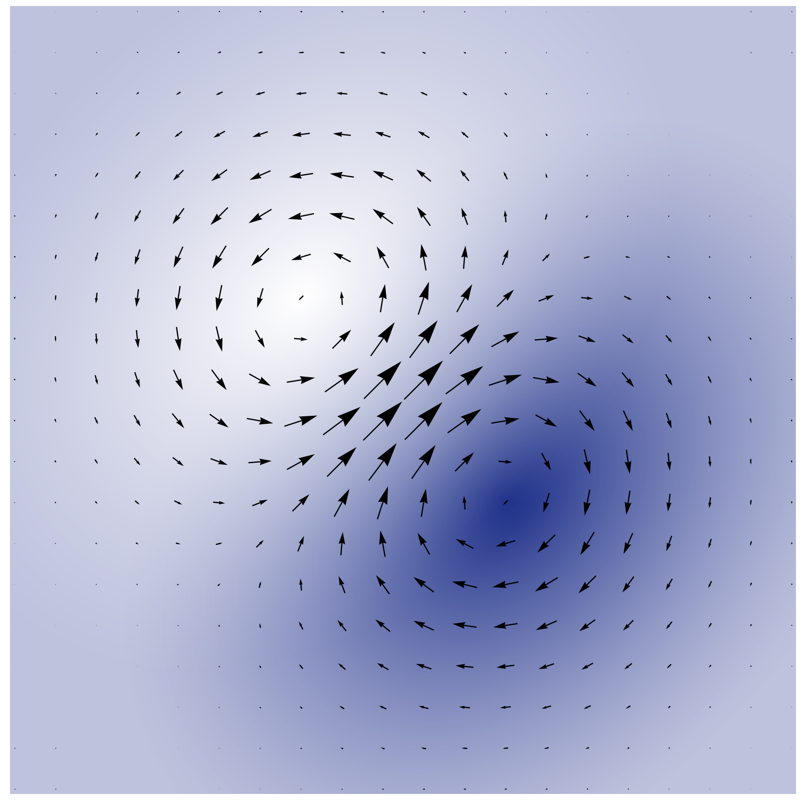
(Discrete) Exterior Calculus

(Discrete) Exterior Calculus

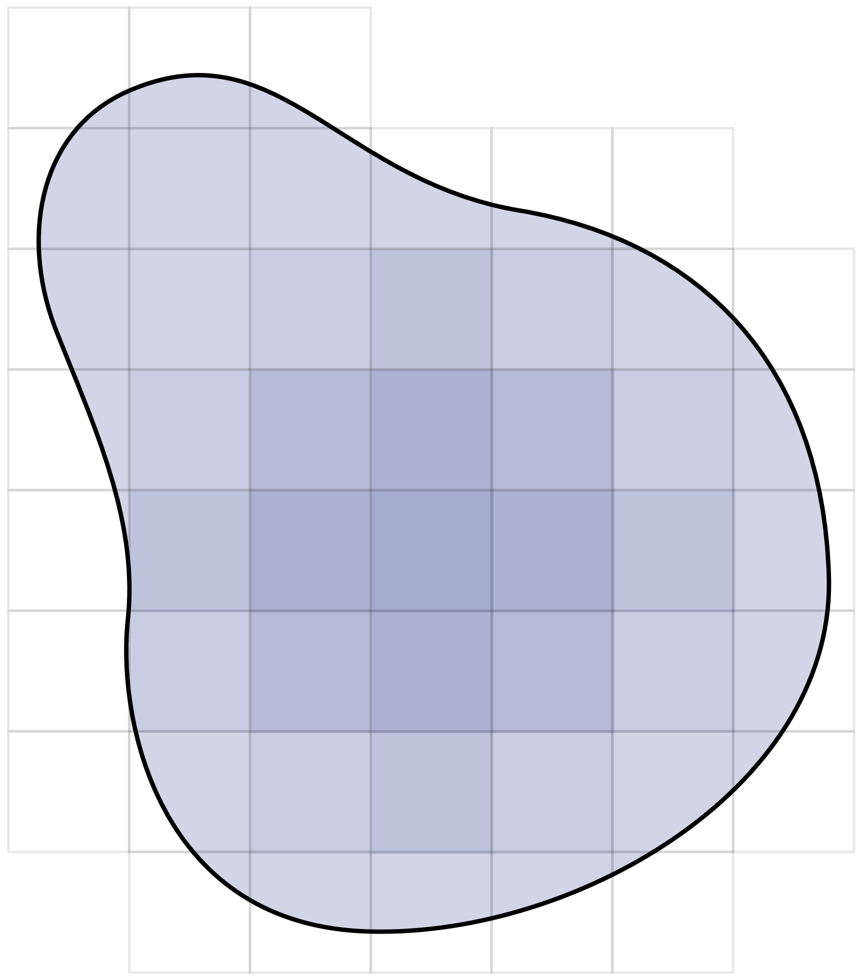
- Exterior algebra: language of volumes/measurements

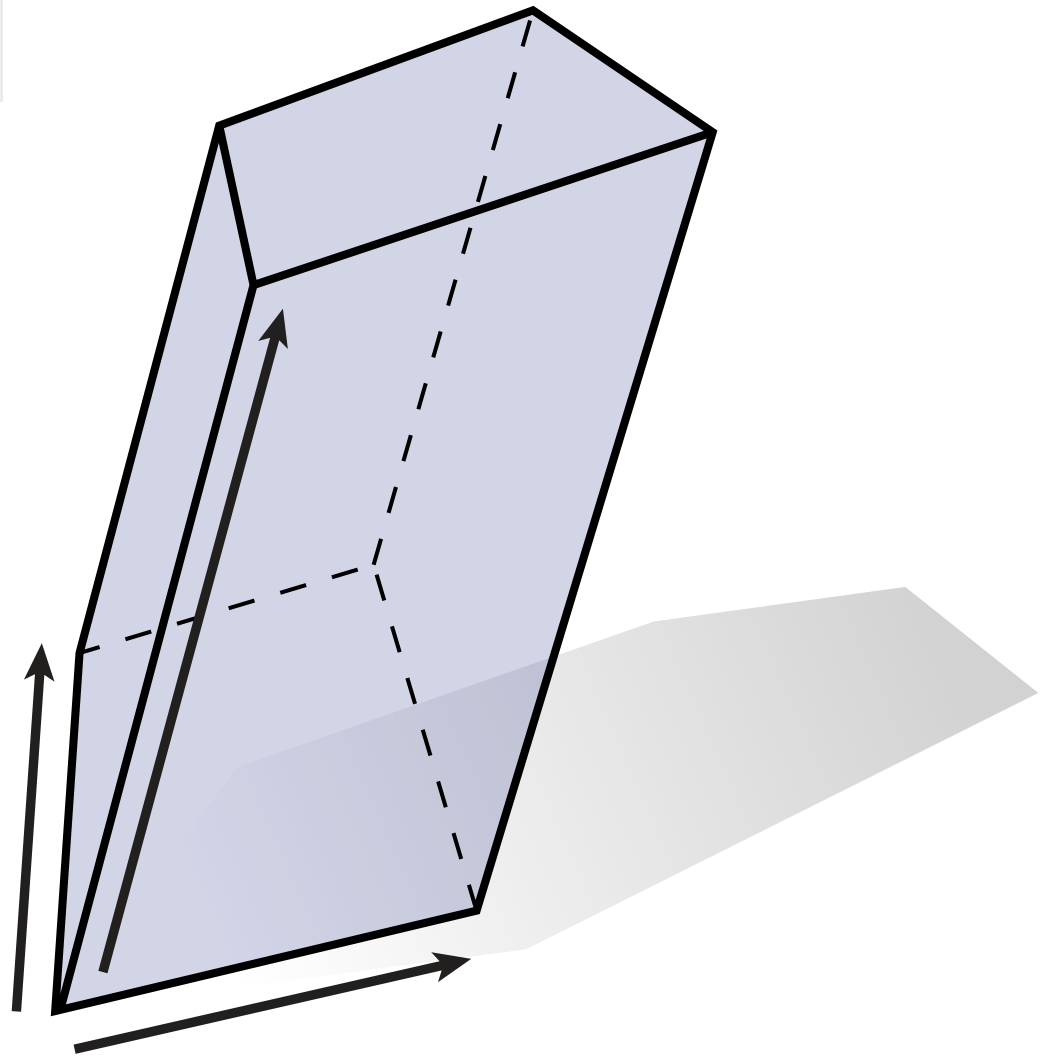


(Discrete) Exterior Calculus



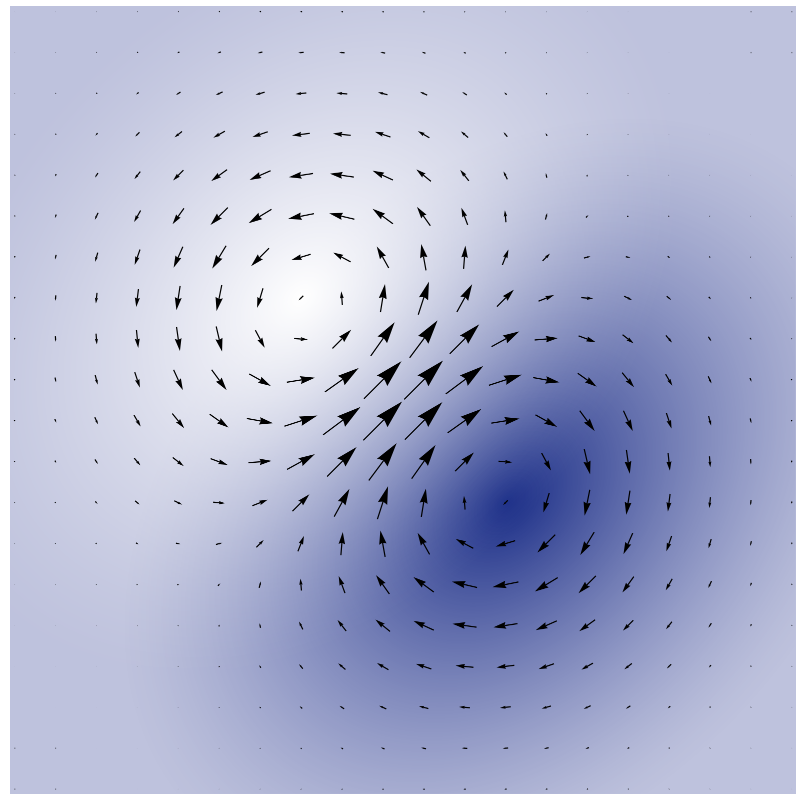
- Exterior algebra: language of volumes/measurements
- Exterior calculus: differentiate, integrate

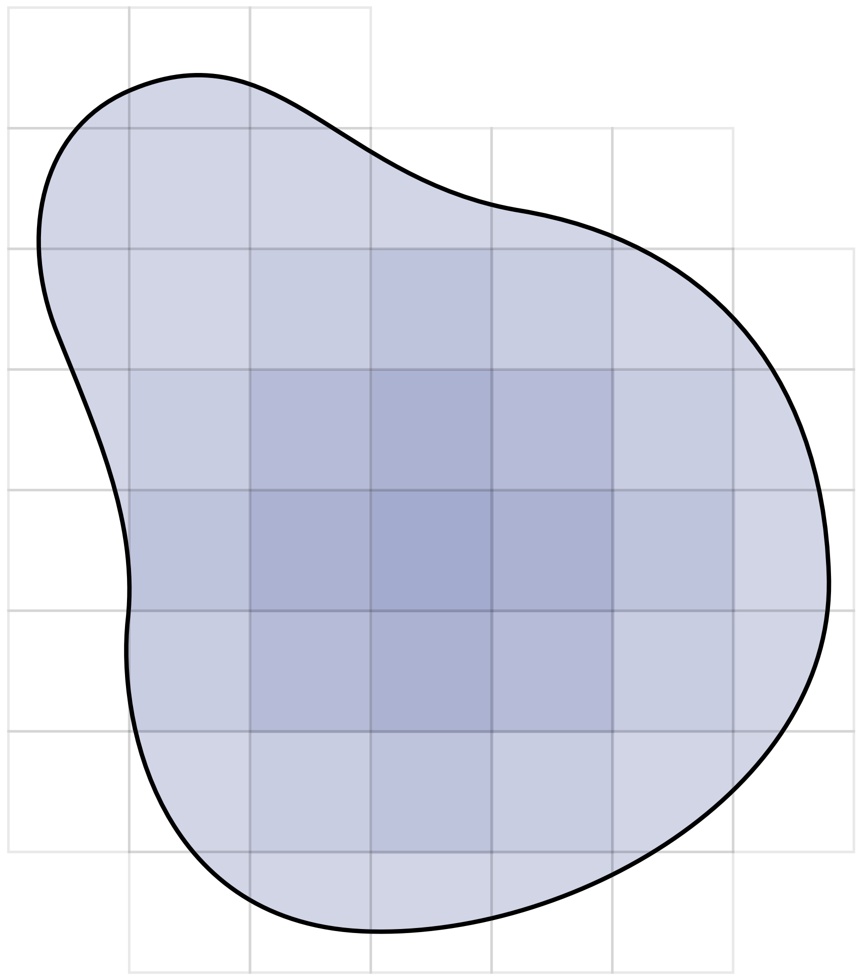


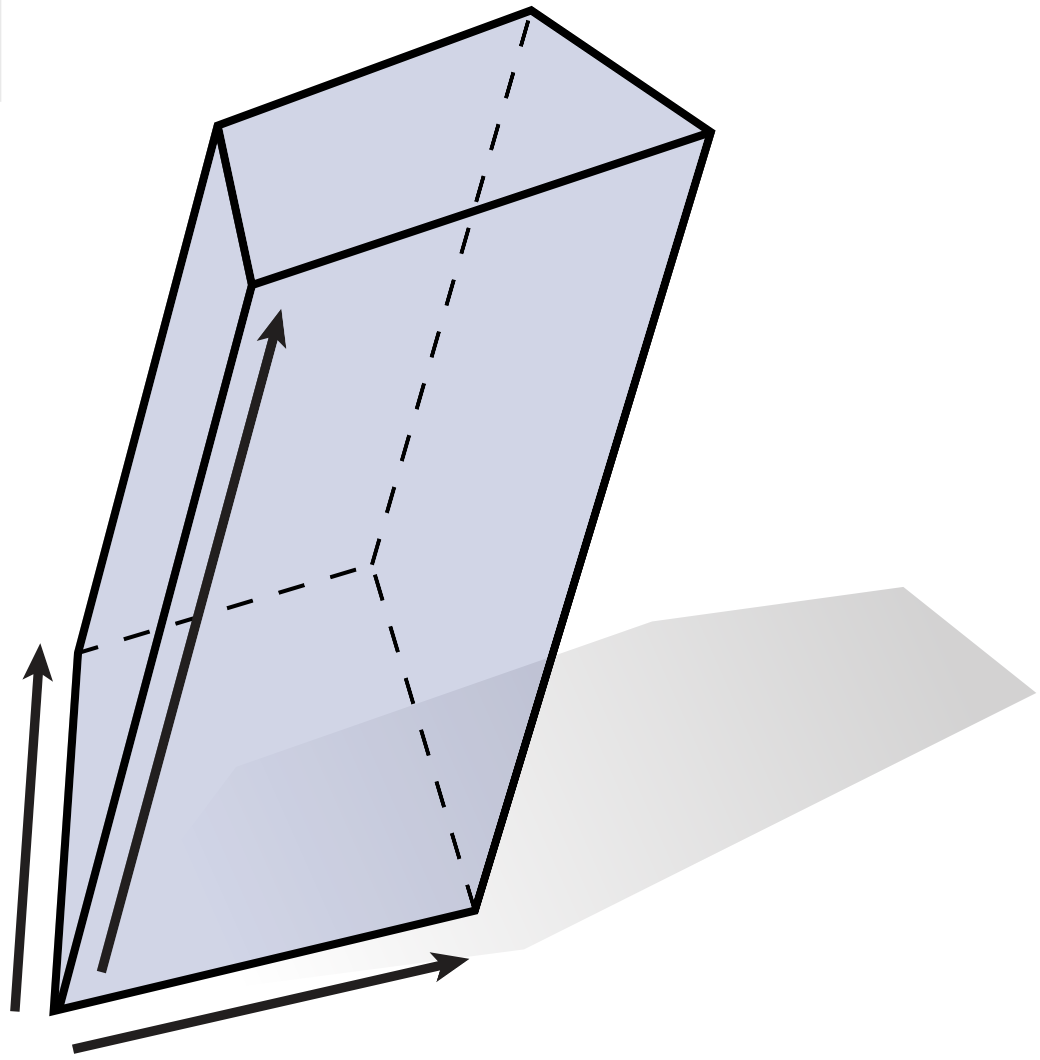


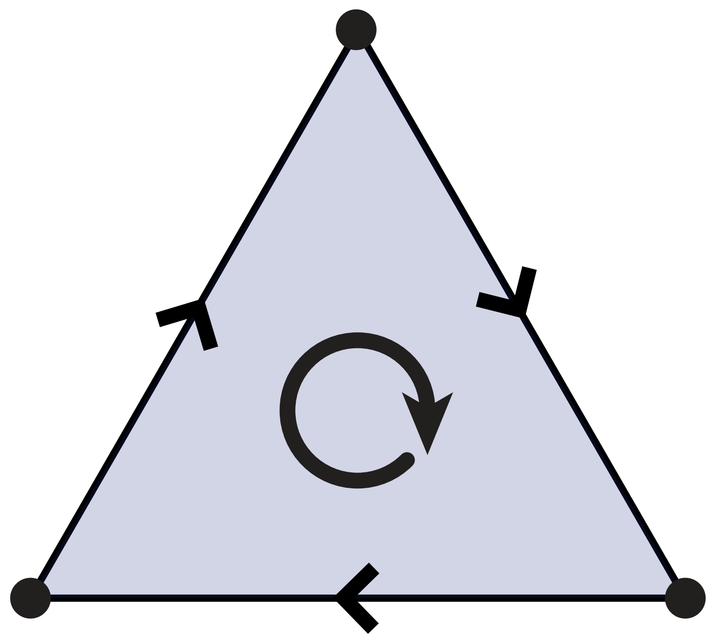
(Discrete) Exterior Calculus

- Exterior algebra: language of volumes/measurements
- Exterior calculus: differentiate, integrate
- Discretize? Integrate, Stokes'



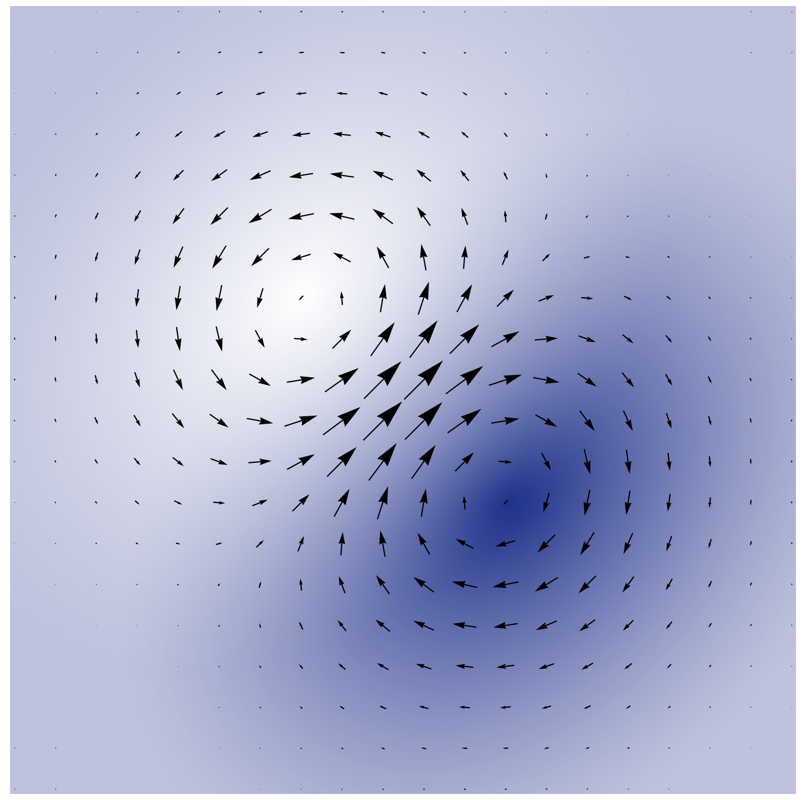


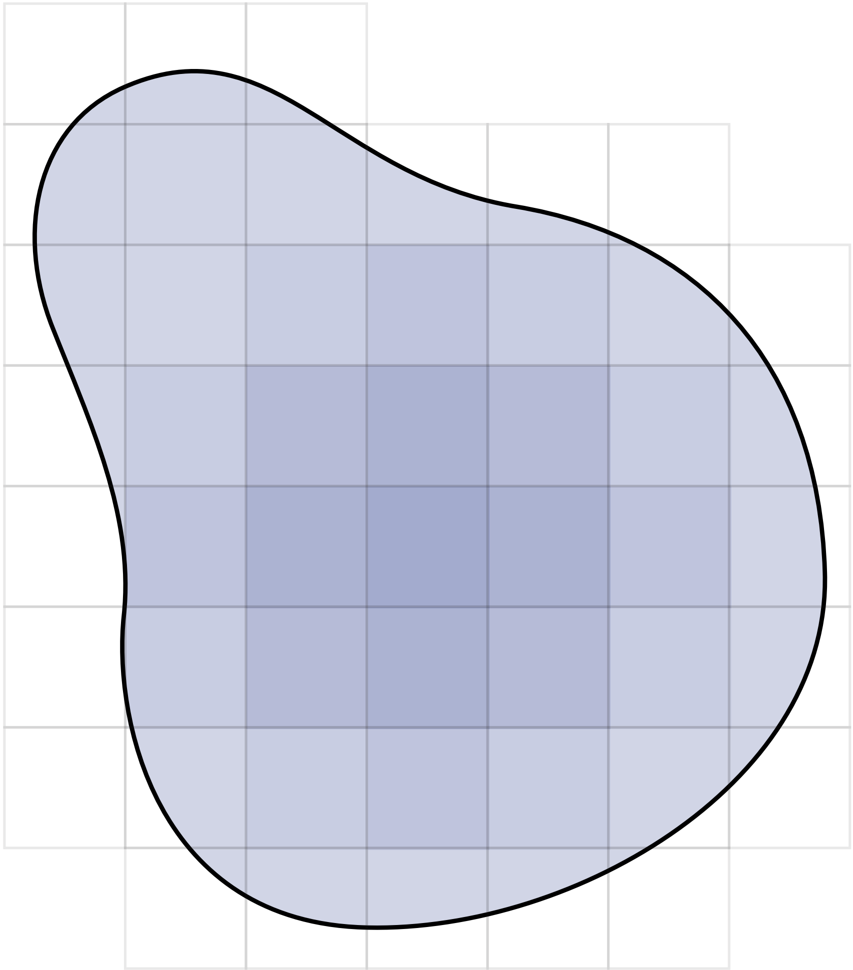


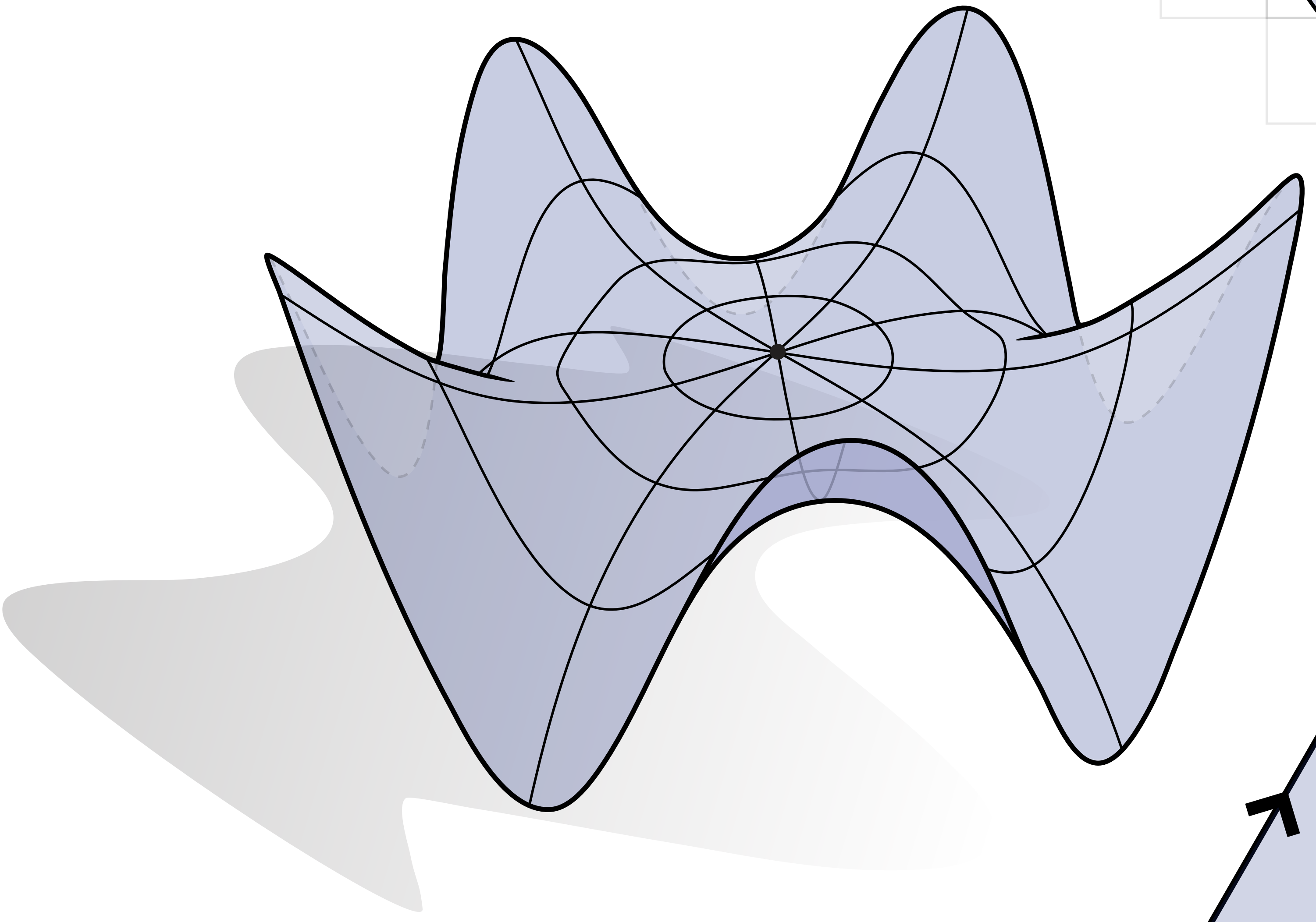


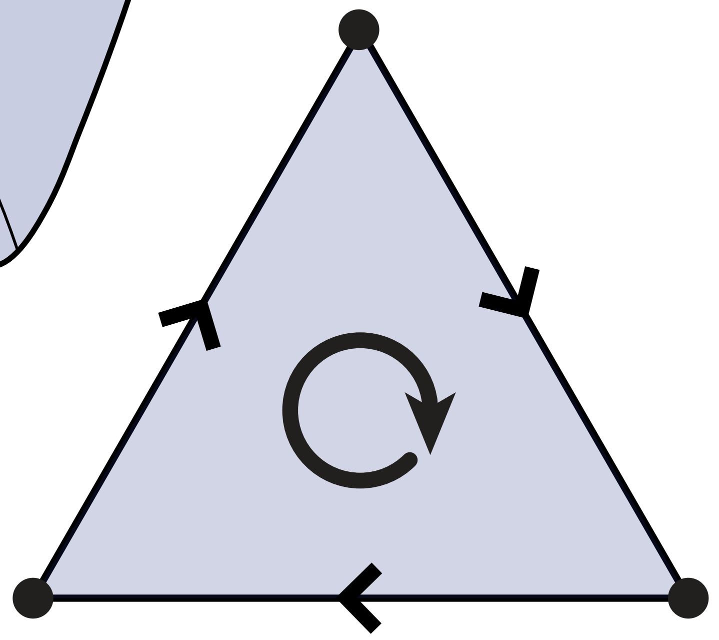
(Discrete) Exterior Calculus

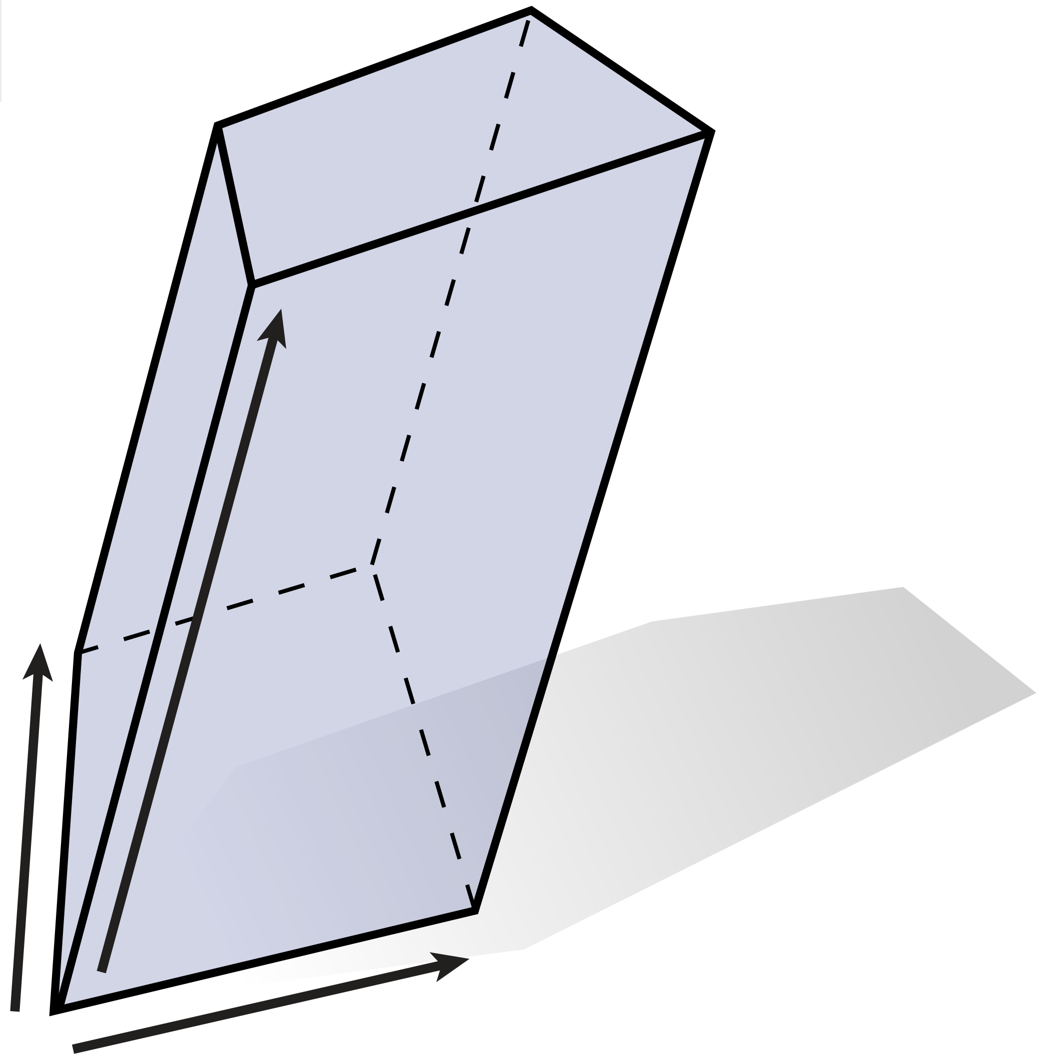
- Exterior algebra: language of volumes/measurements
- Exterior calculus: differentiate, integrate
- Discretize? Integrate, Stokes'
- Poisson equation





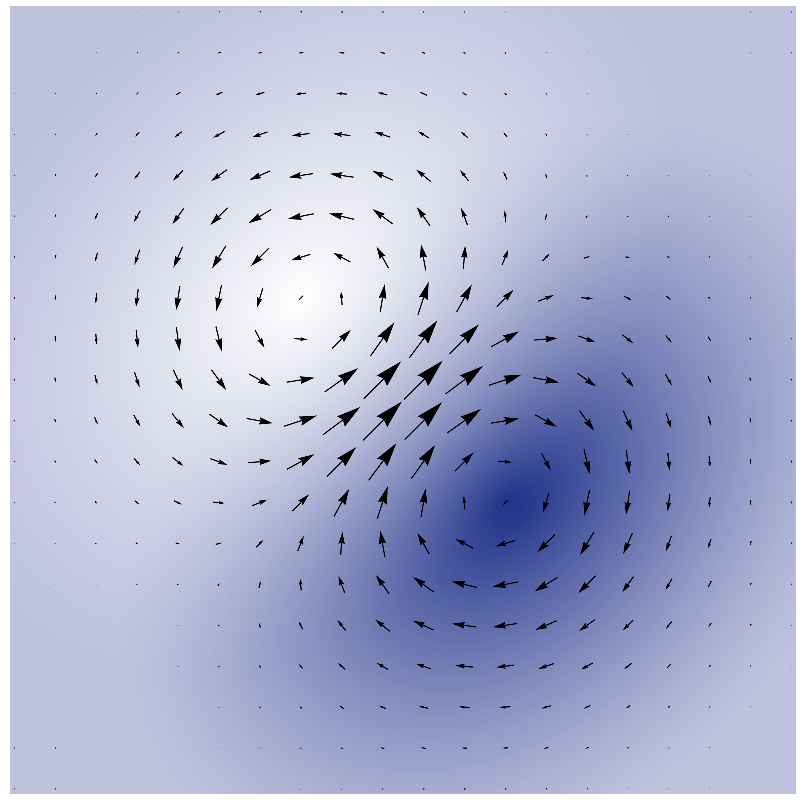
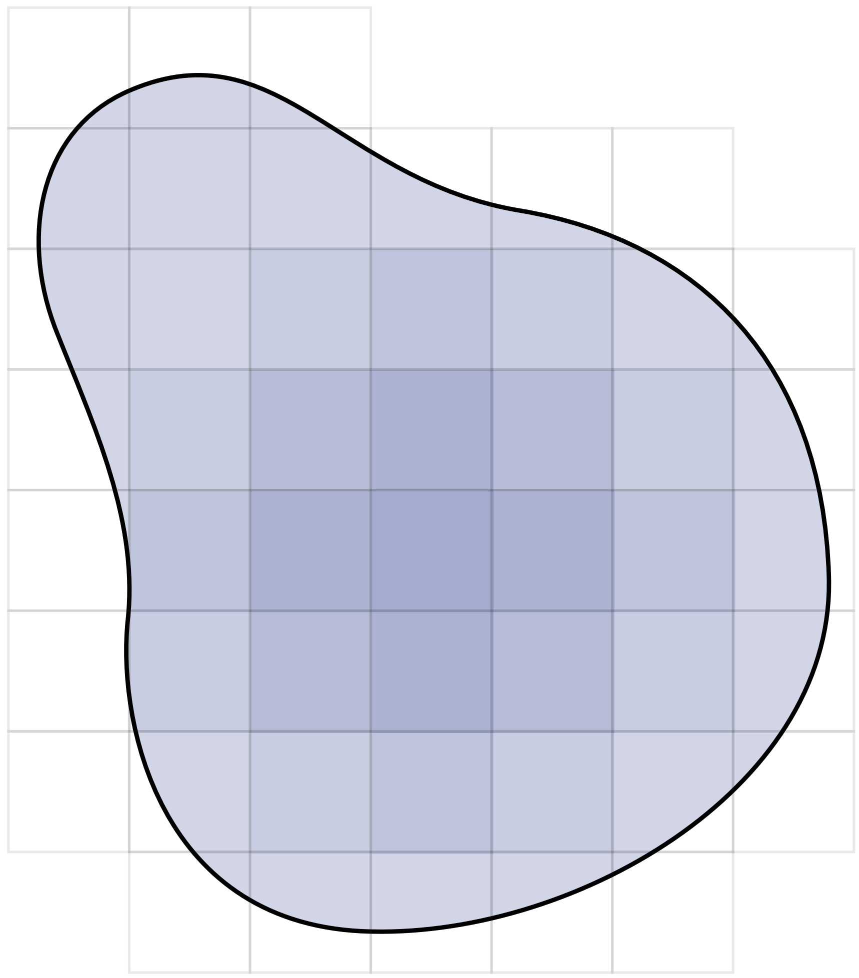
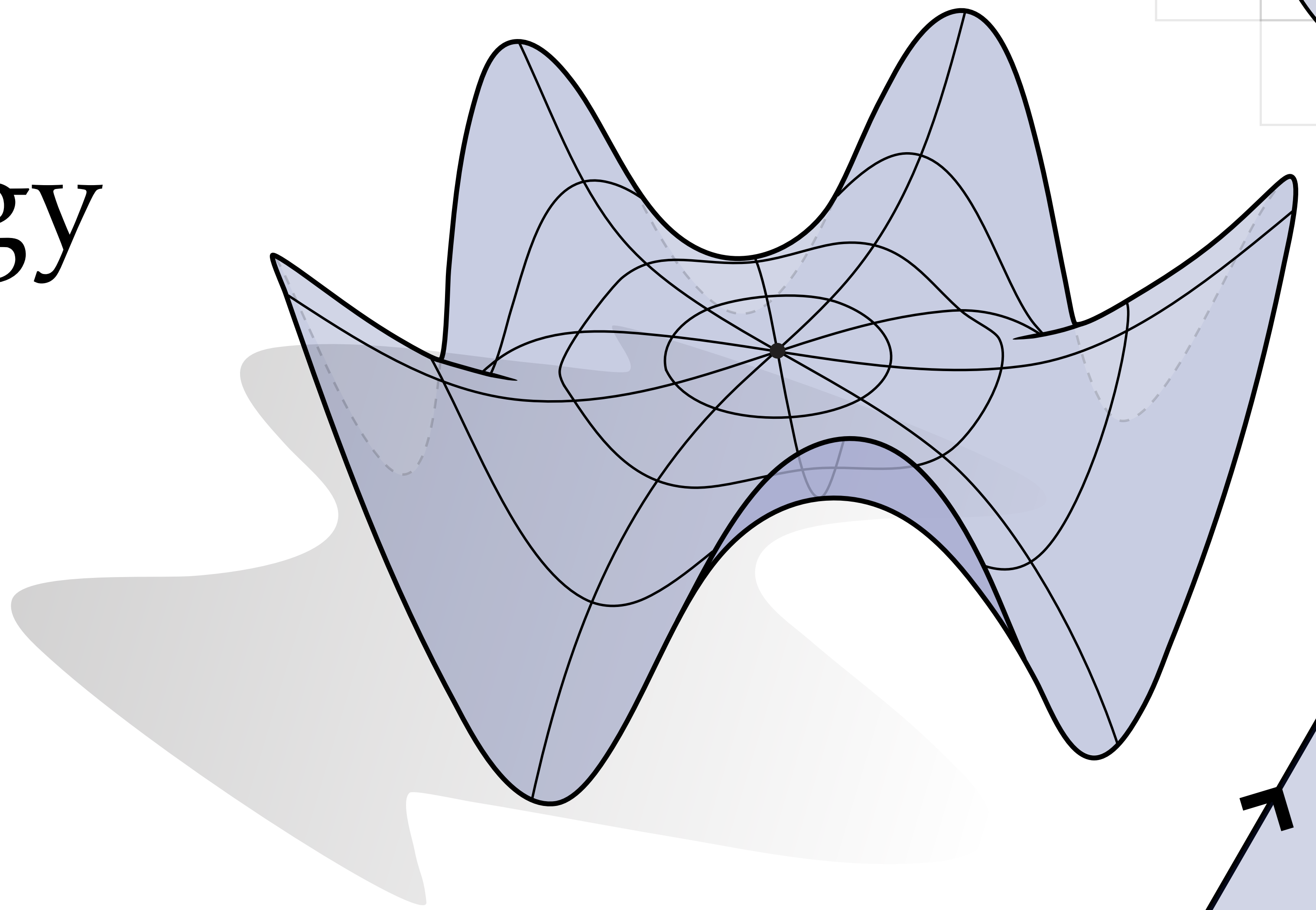
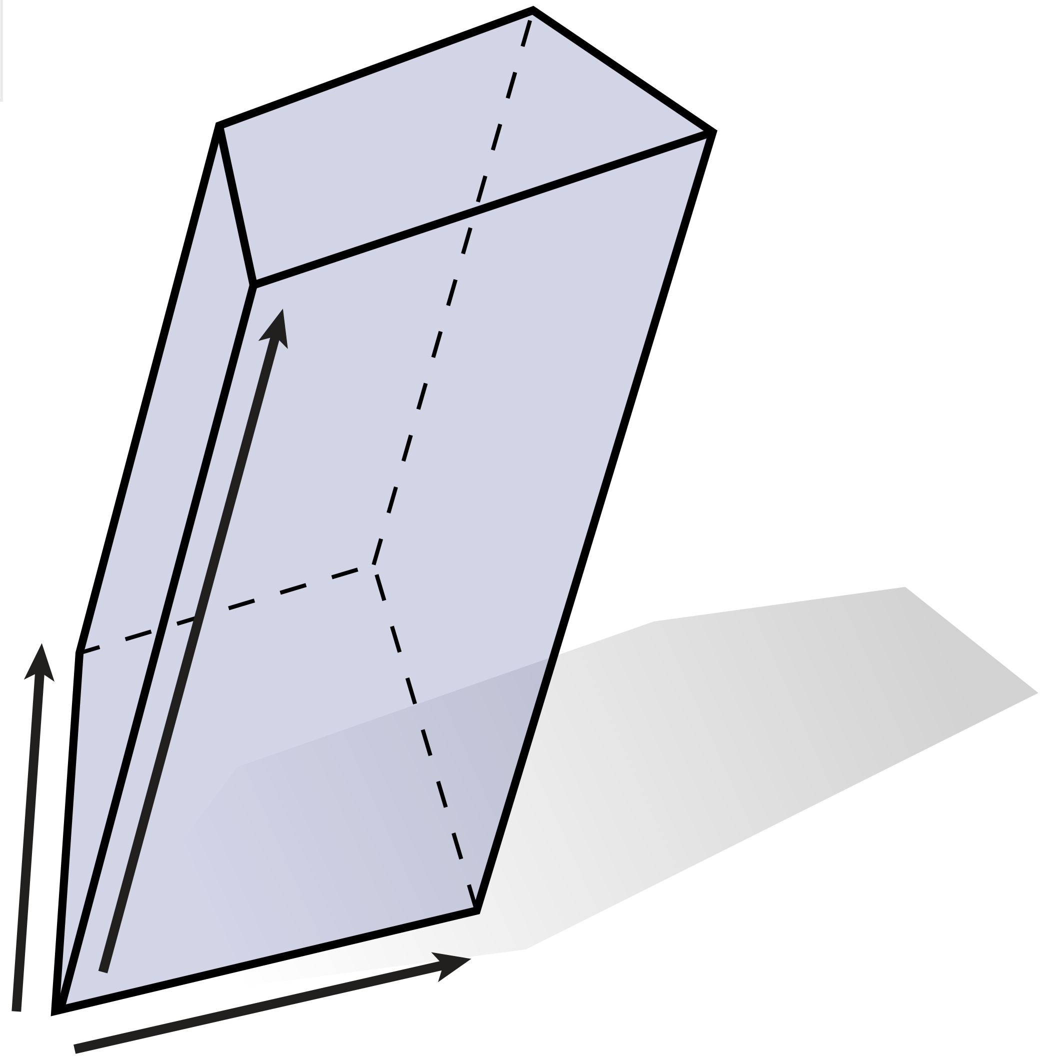
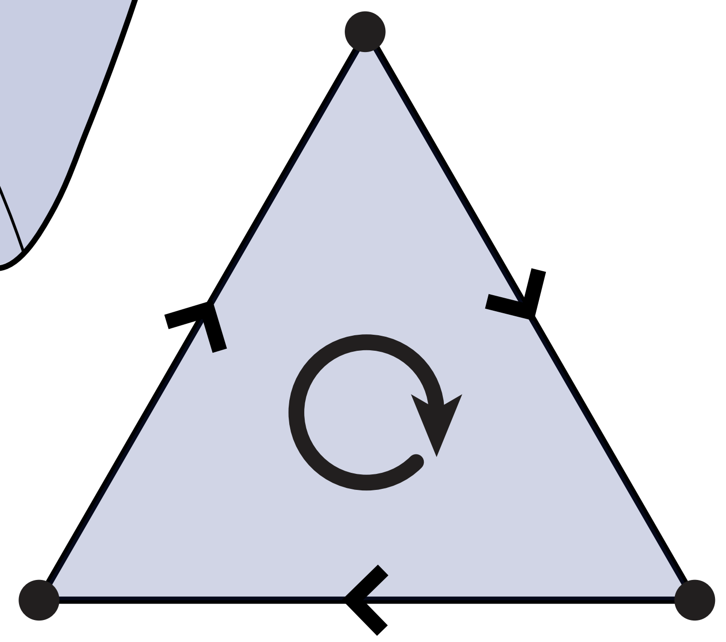
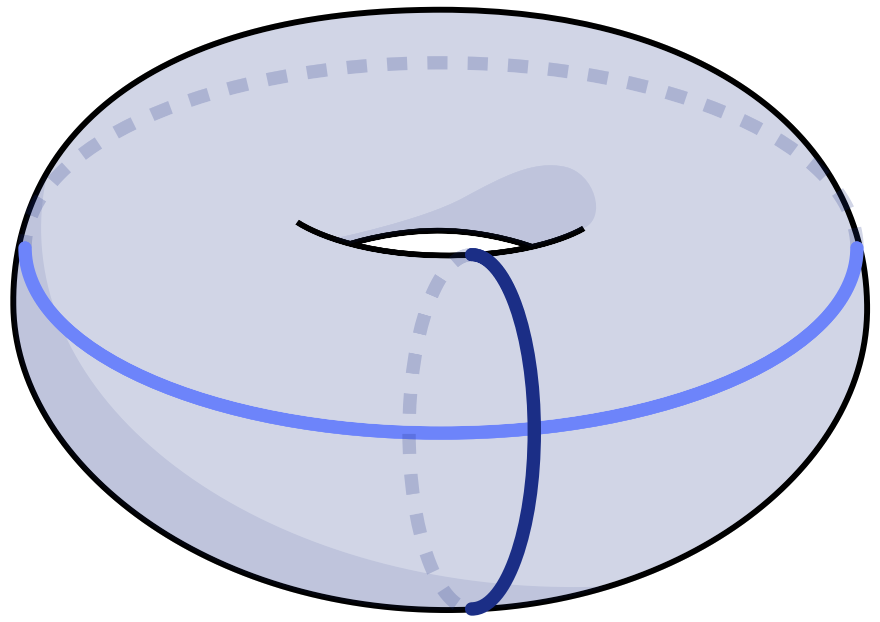
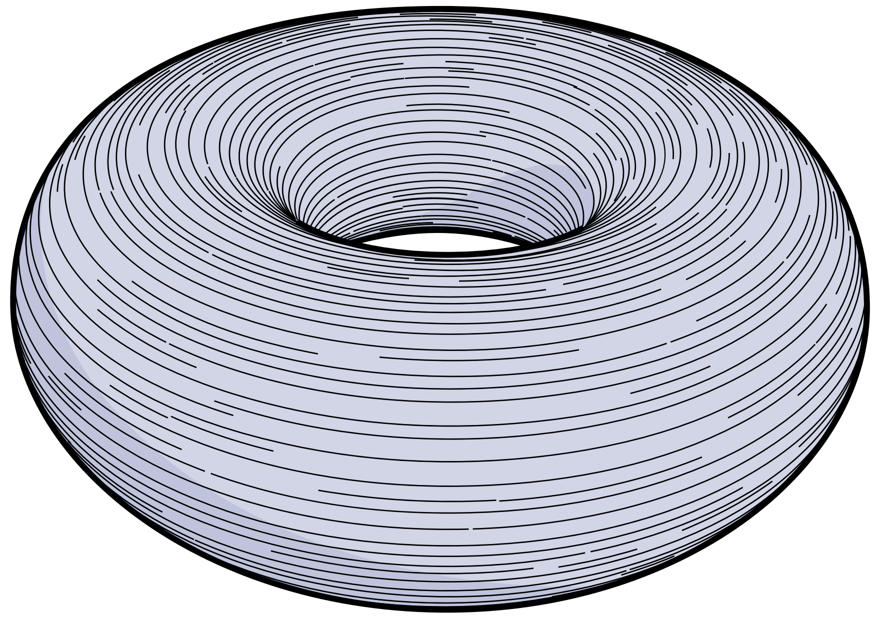




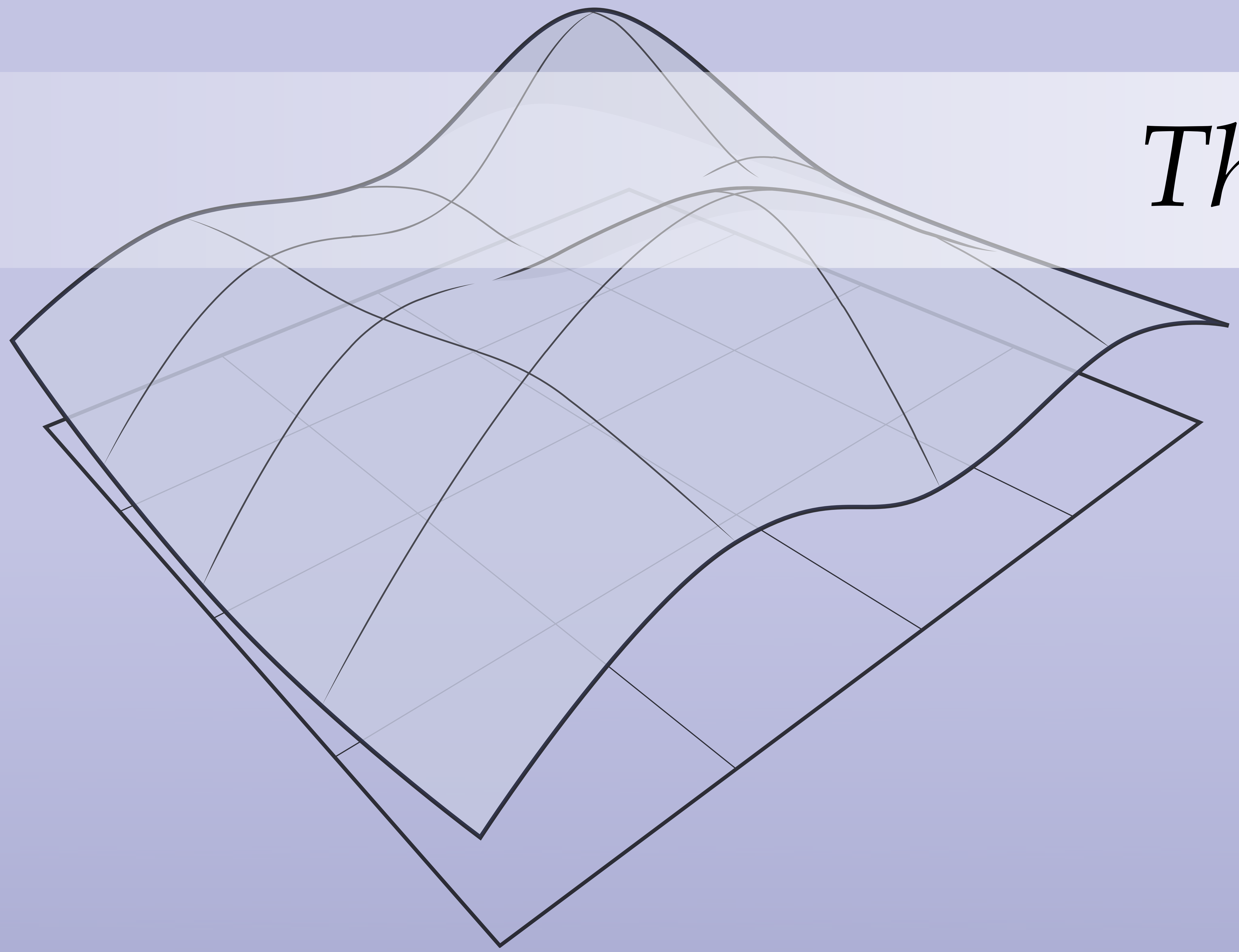


(Discrete) Exterior Calculus

- Exterior algebra: language of volumes/measurements
- Exterior calculus: differentiate, integrate
- Discretize? Integrate, Stokes'
- Poisson equation
- Homology & cohomology

Next: Applications!



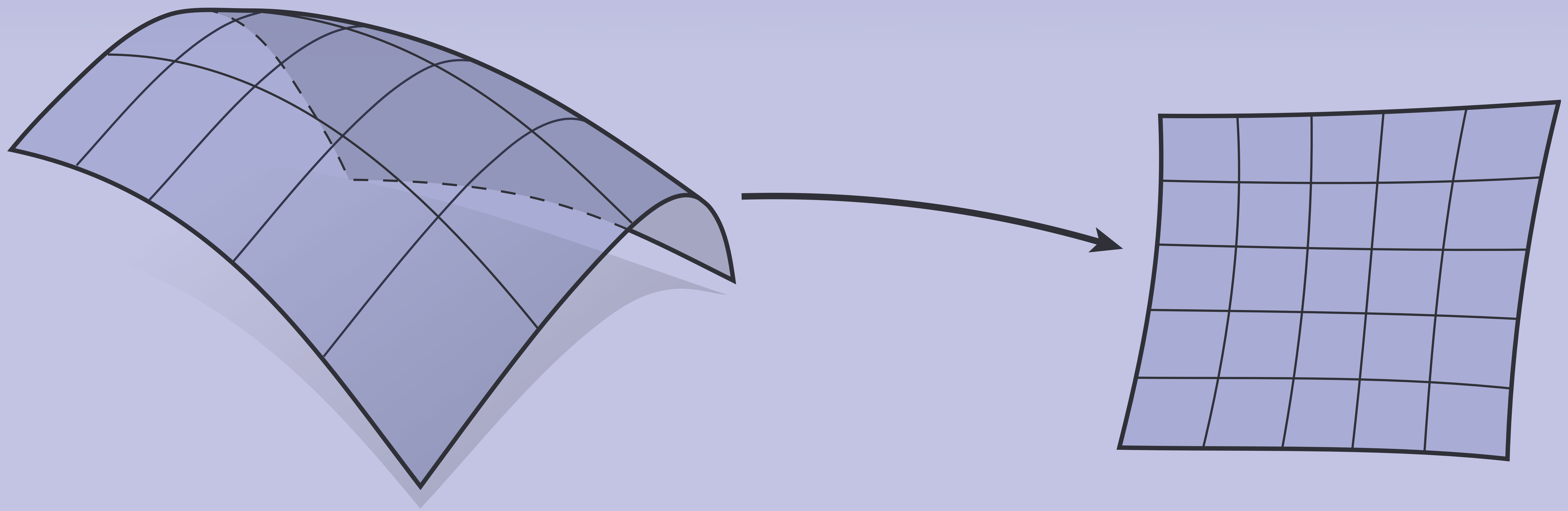
Thanks!

DIGITAL GEOMETRY PROCESSING
WITH DISCRETE EXTERIOR CALCULUS

Keenan Crane • Fernando de Goes • Mathieu Desbrun • Peter Schröder

SIGGRAPH 2013

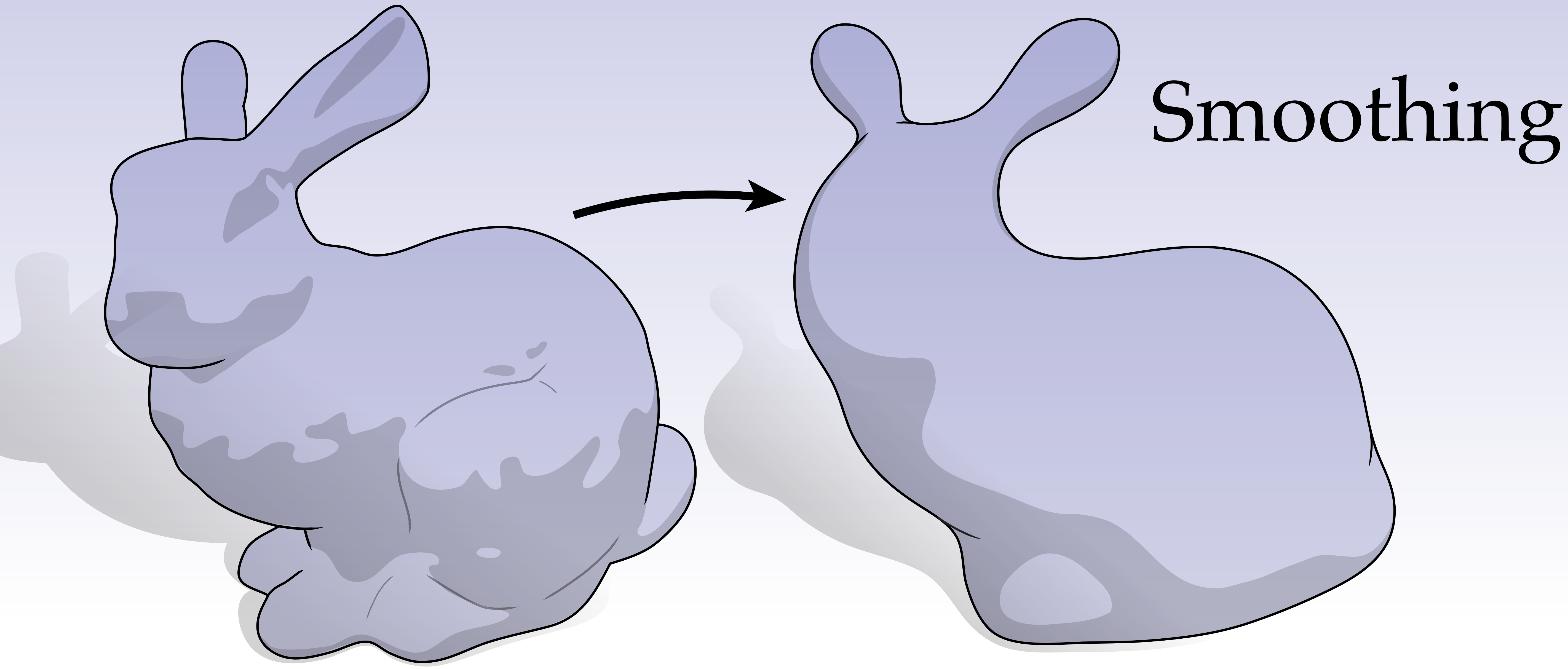
Part III: Implementation & Applications



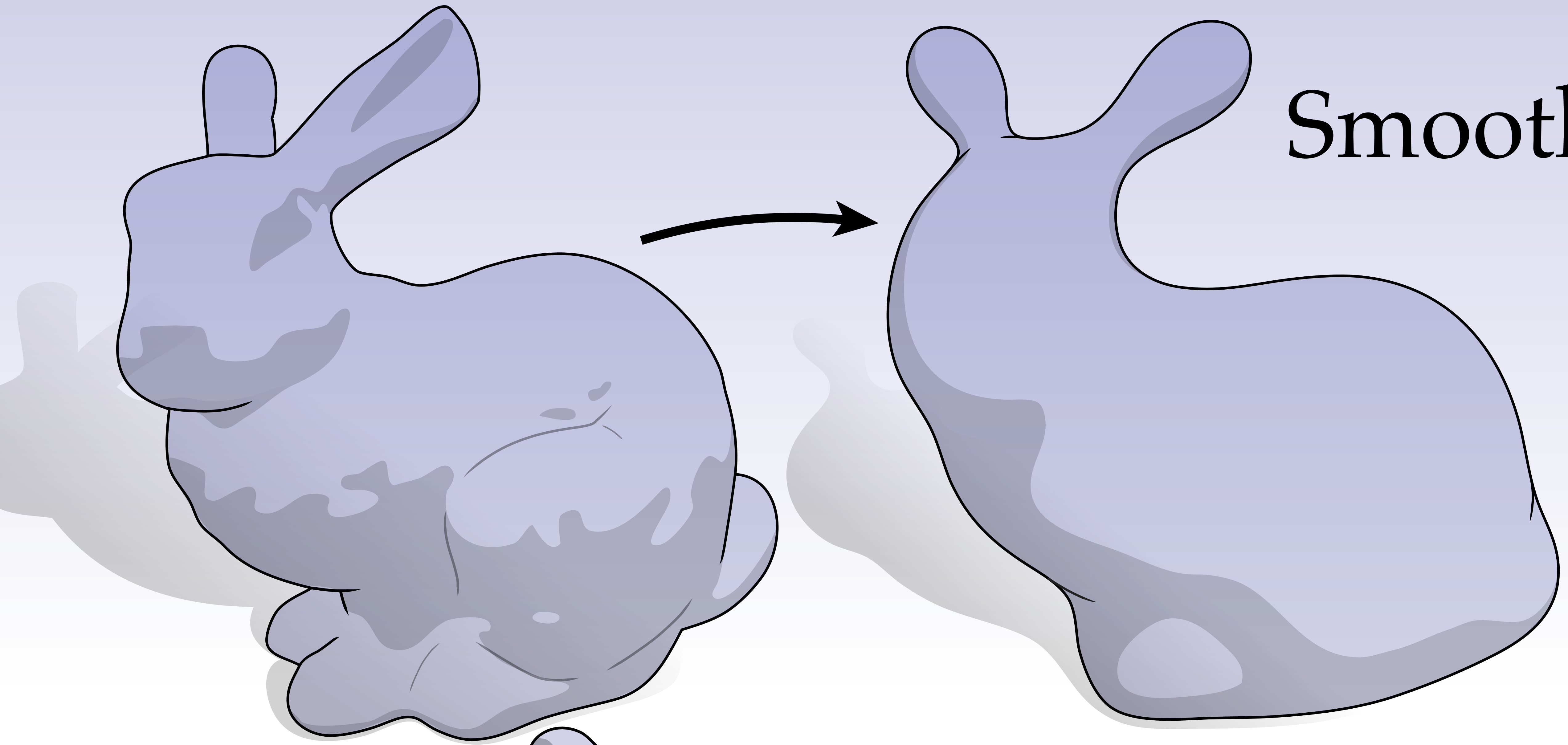
Digital Geometry Processing with Discrete Exterior Calculus

Keenan Crane • Fernando de Goes • Mathieu Desbrun • Peter Schröder

Outline

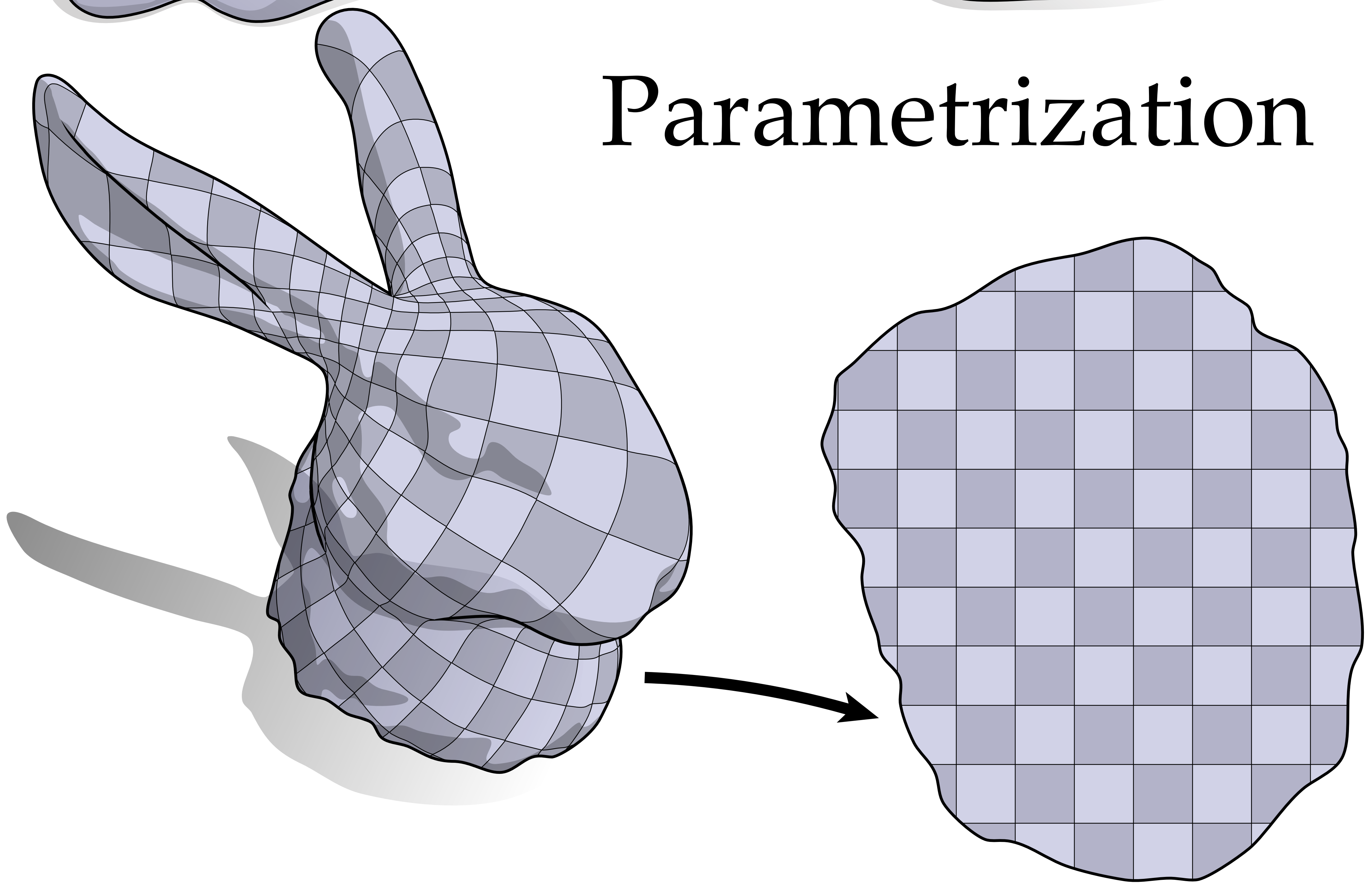


Outline

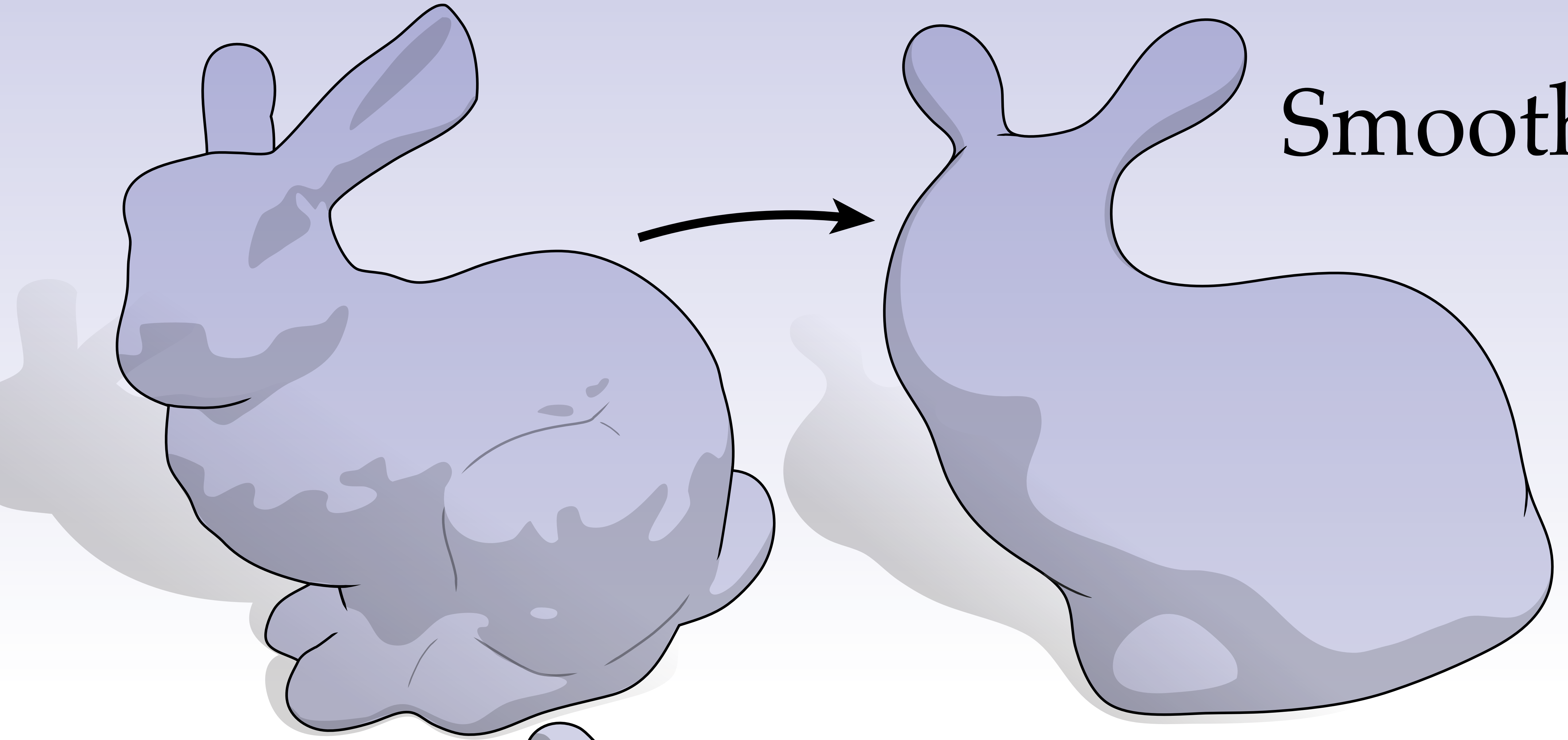


Smoothing

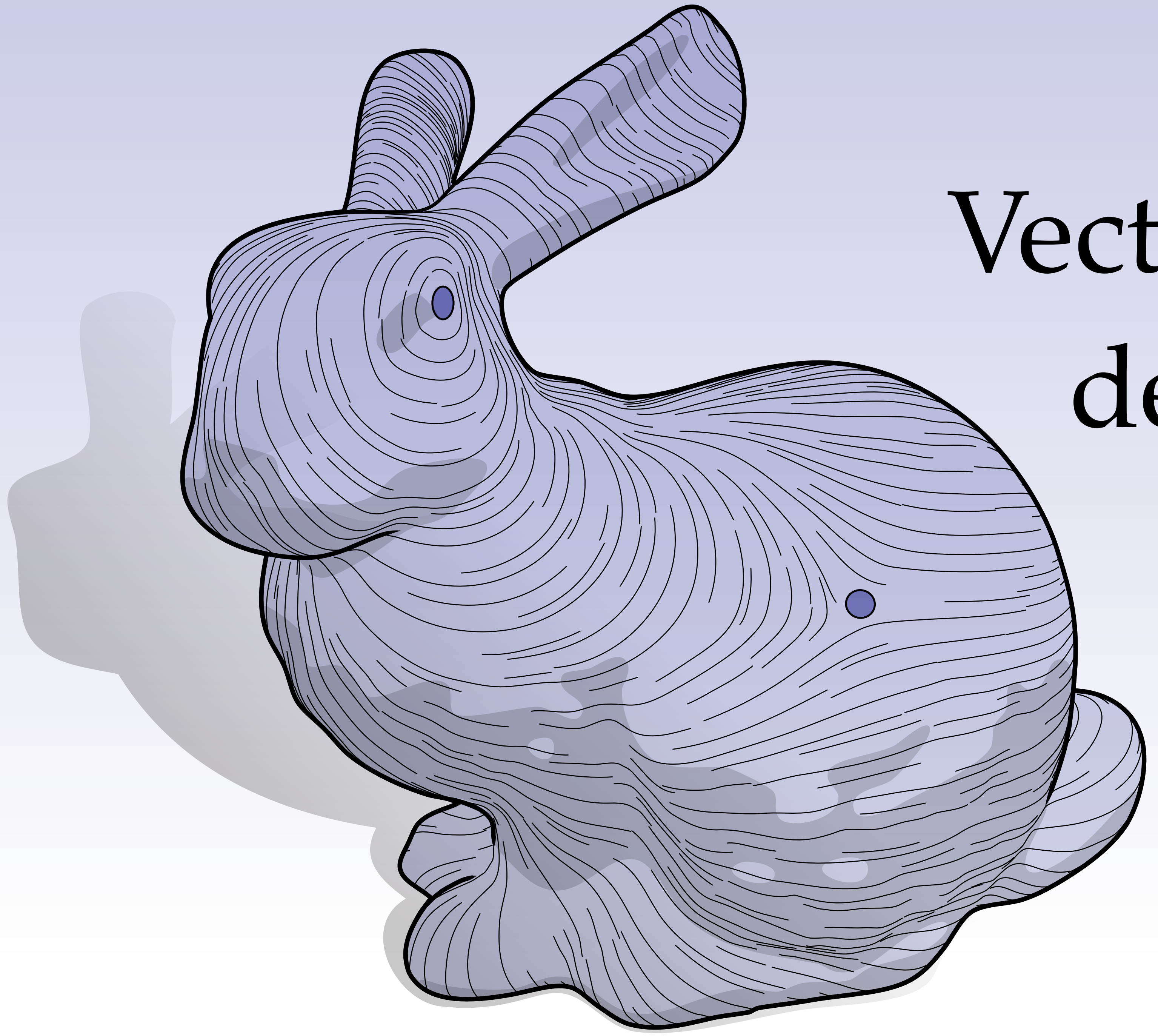
Parametrization



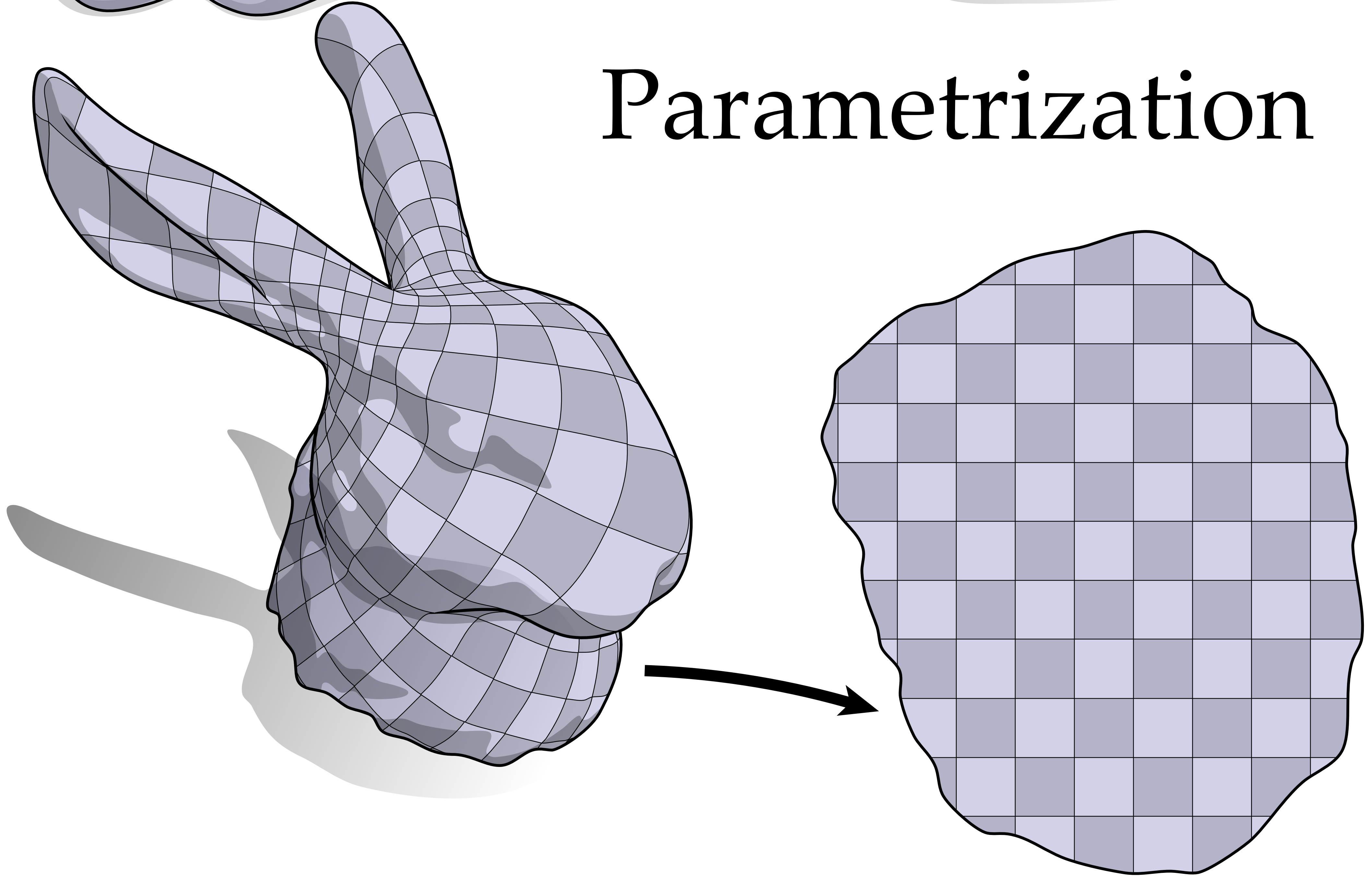
Outline



Smoothing

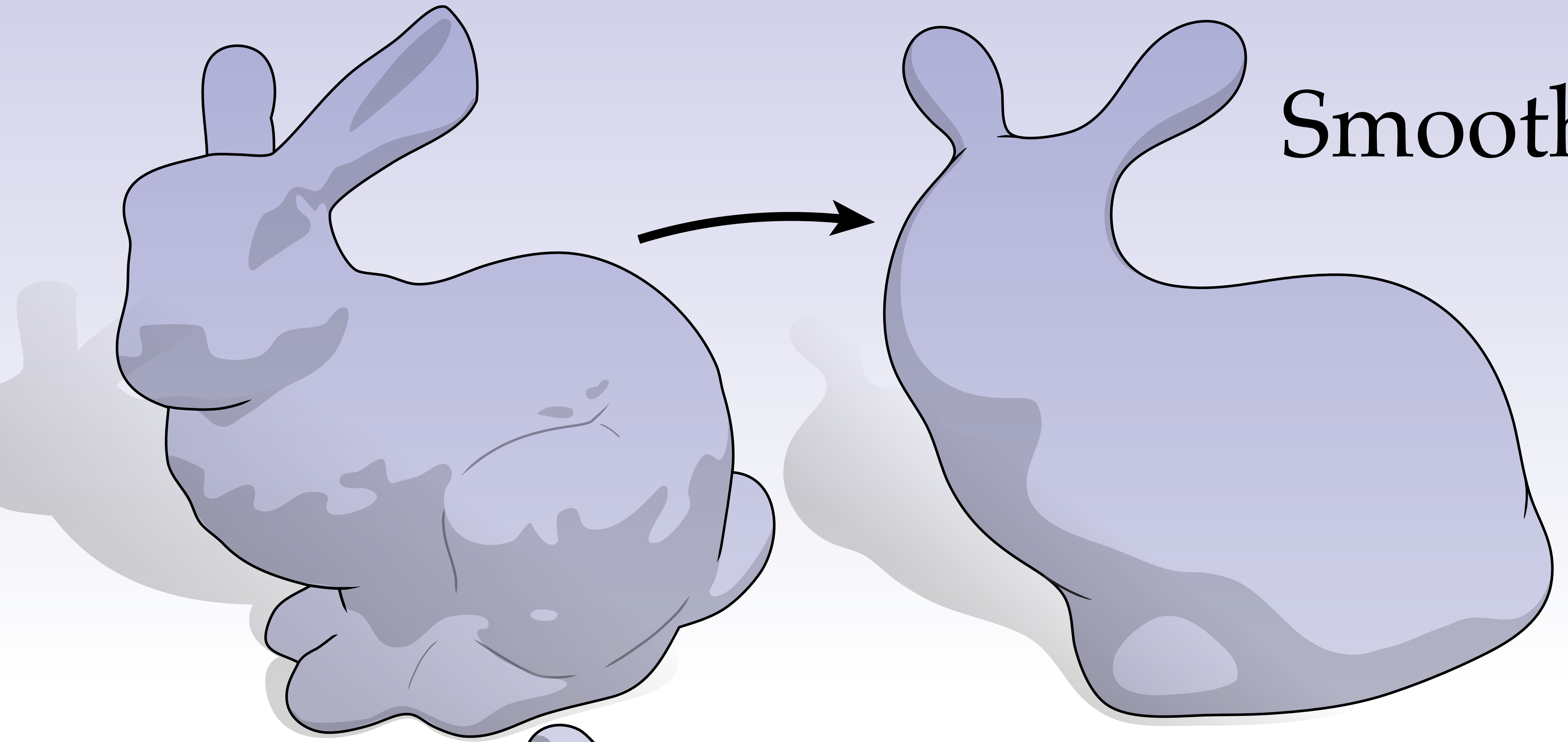


Vector field design

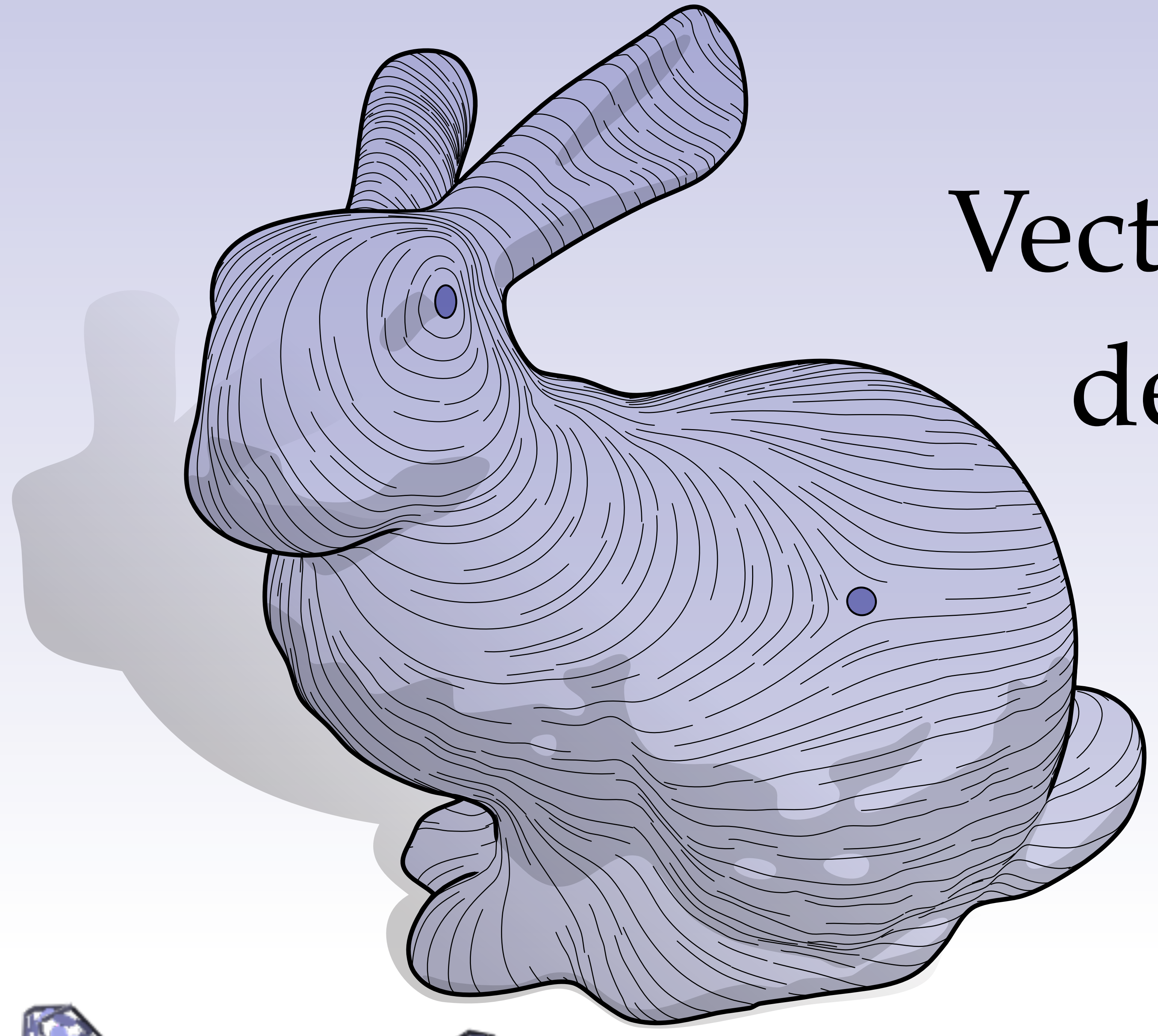


Parametrization

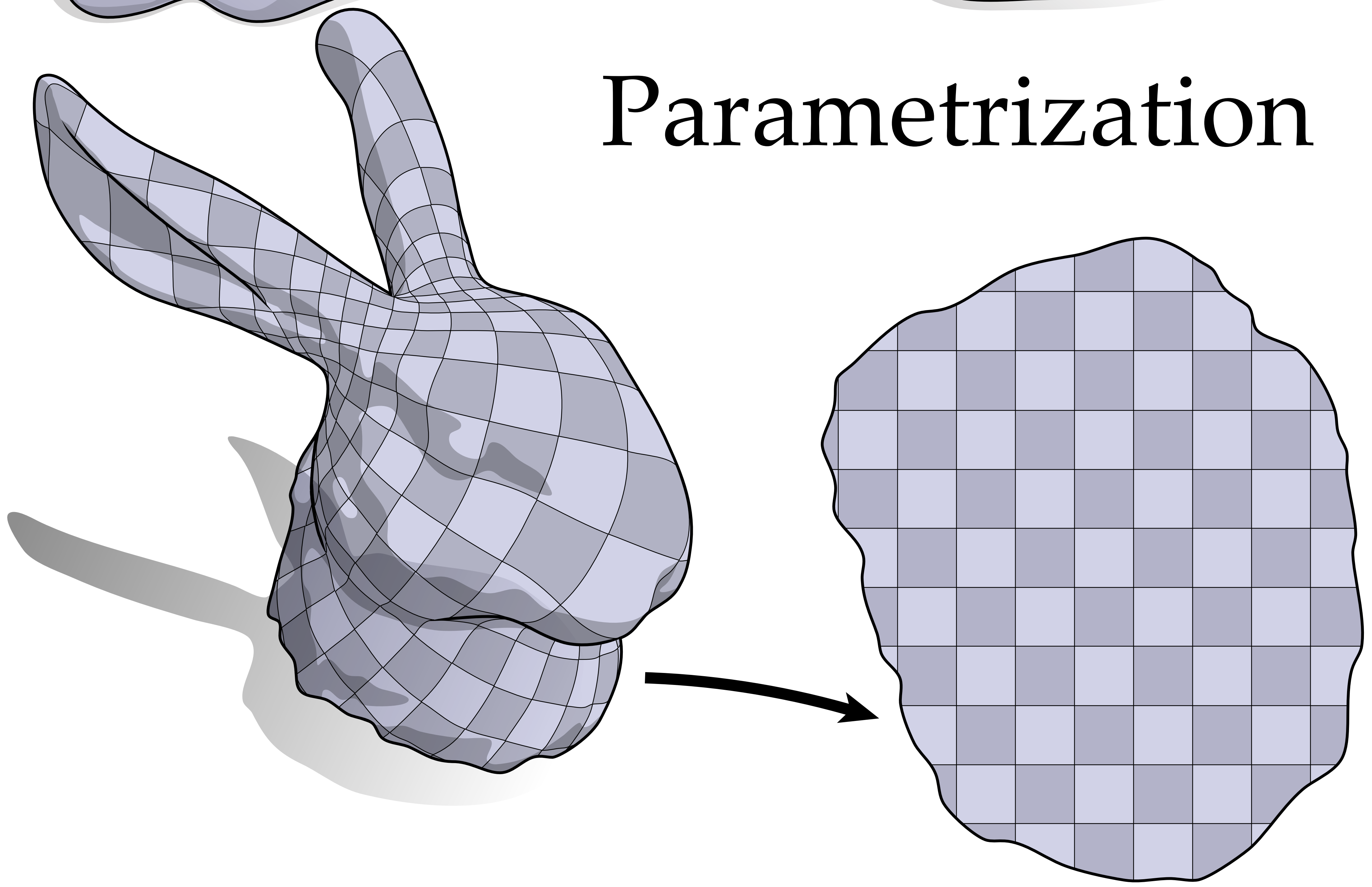
Outline



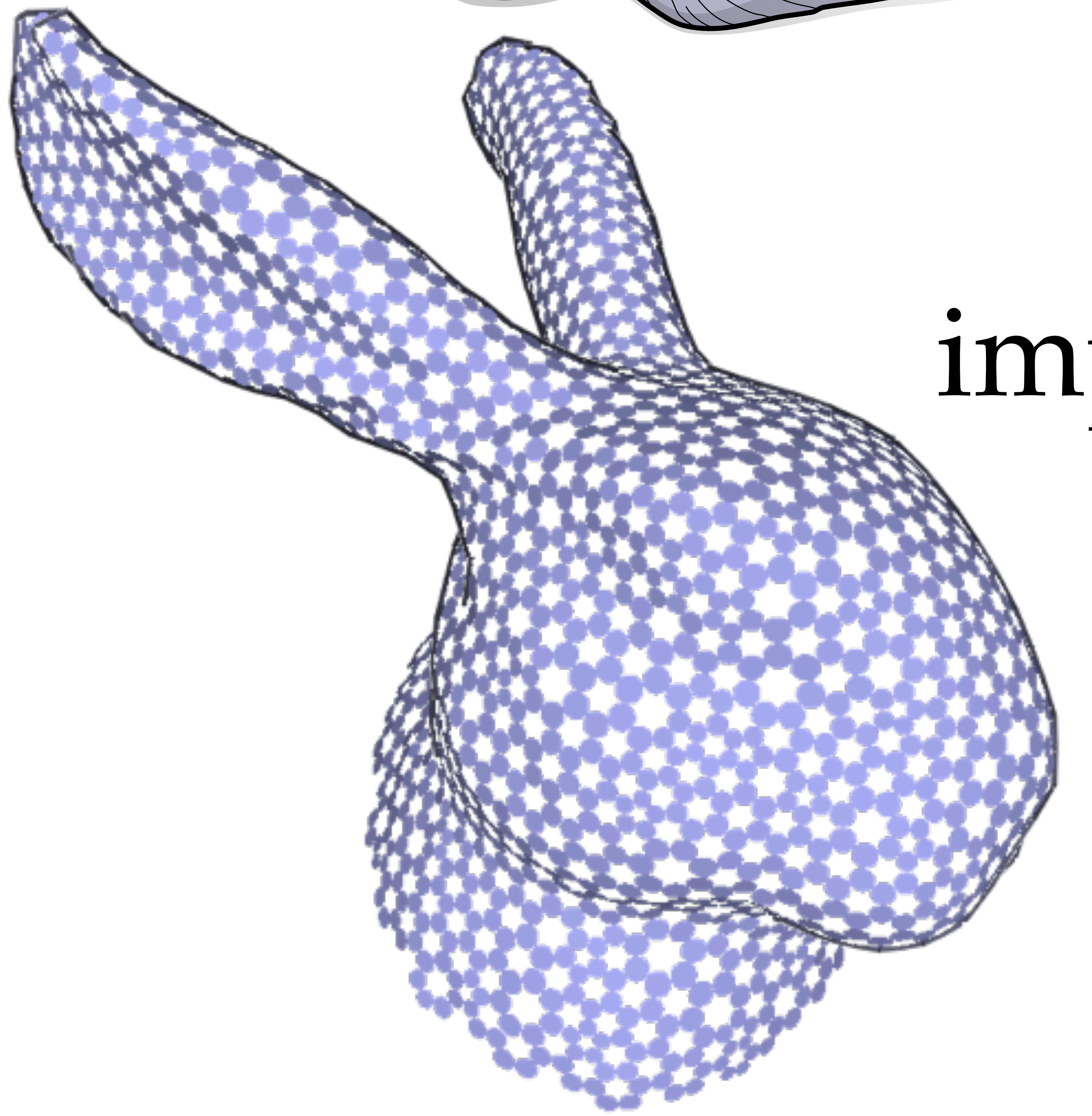
Smoothing



Vector field design



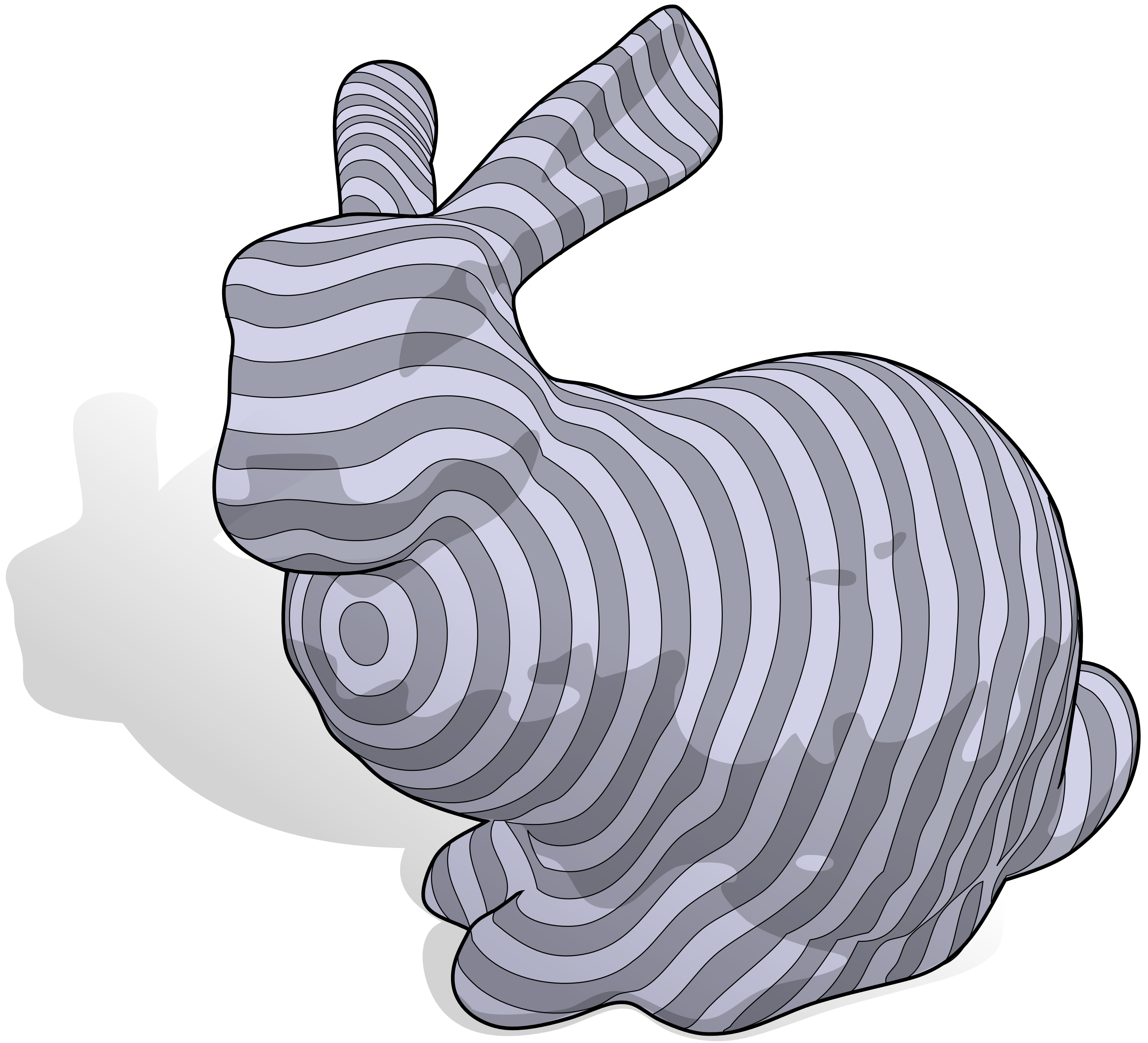
Parametrization



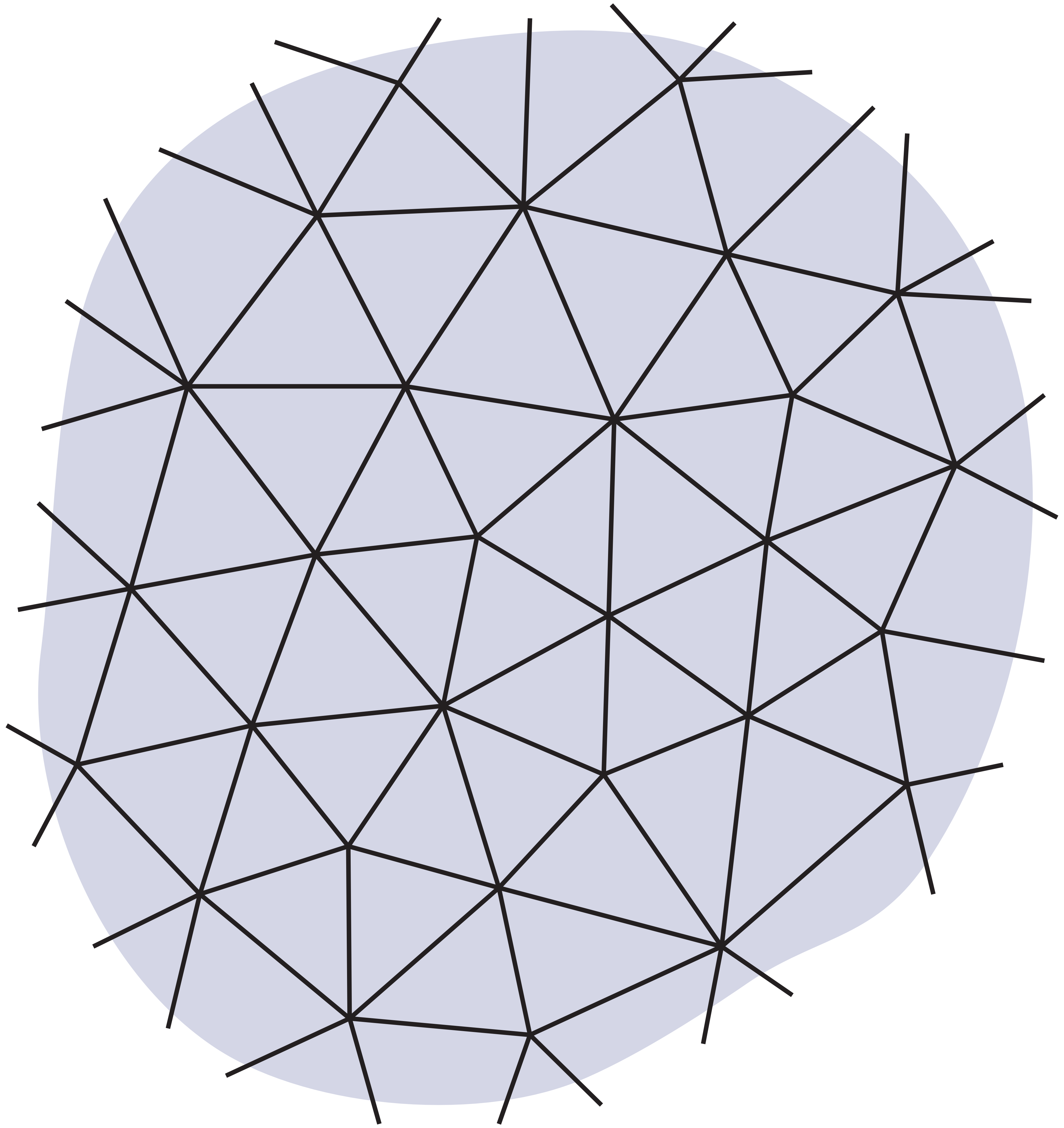
Mesh improvement

github.com/dgpdec/course

github.com/dgpdec/course

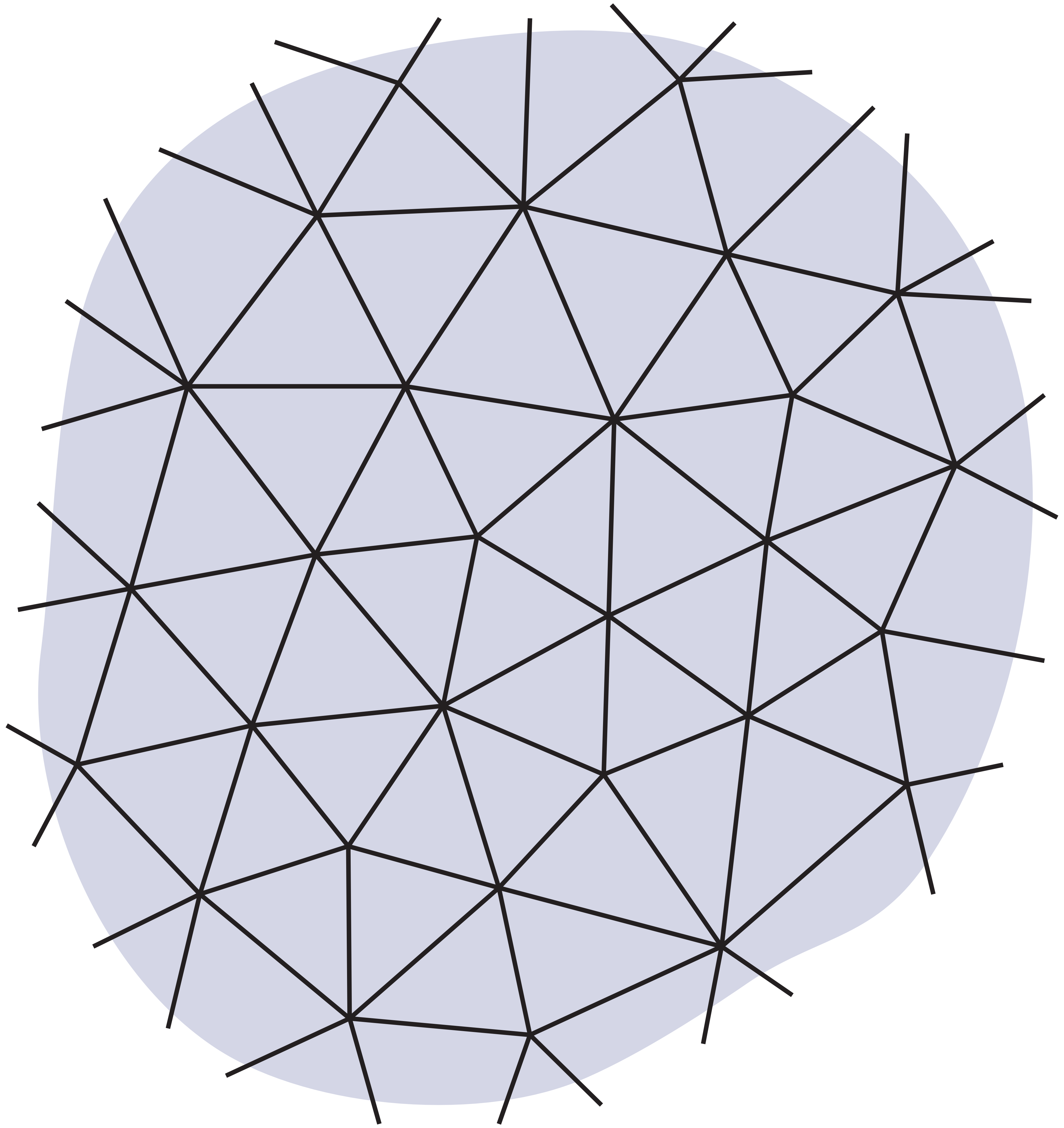


DEC on Triangle Meshes



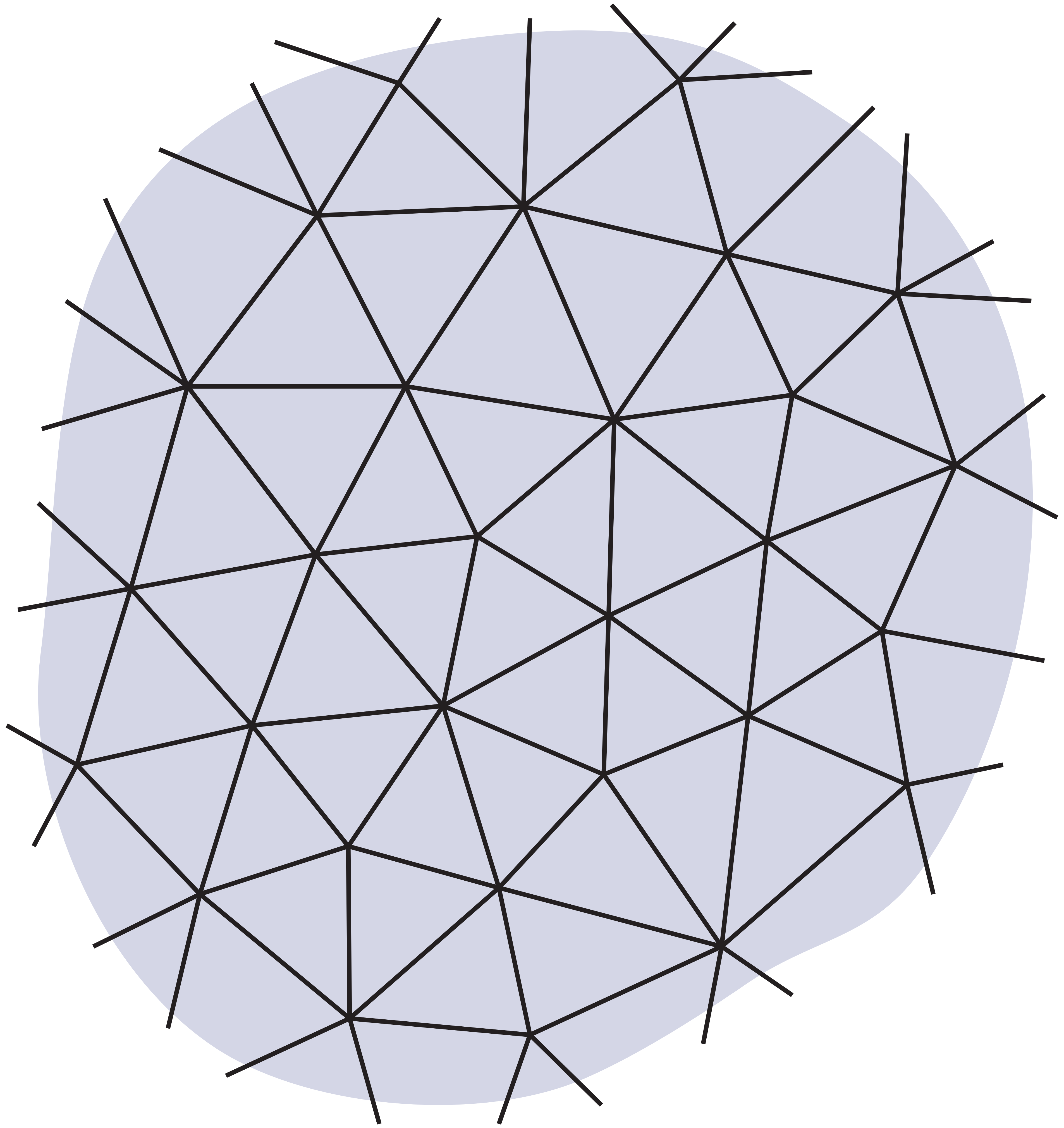
DEC on Triangle Meshes

- Discrete differential forms
  - 0,1,2-forms



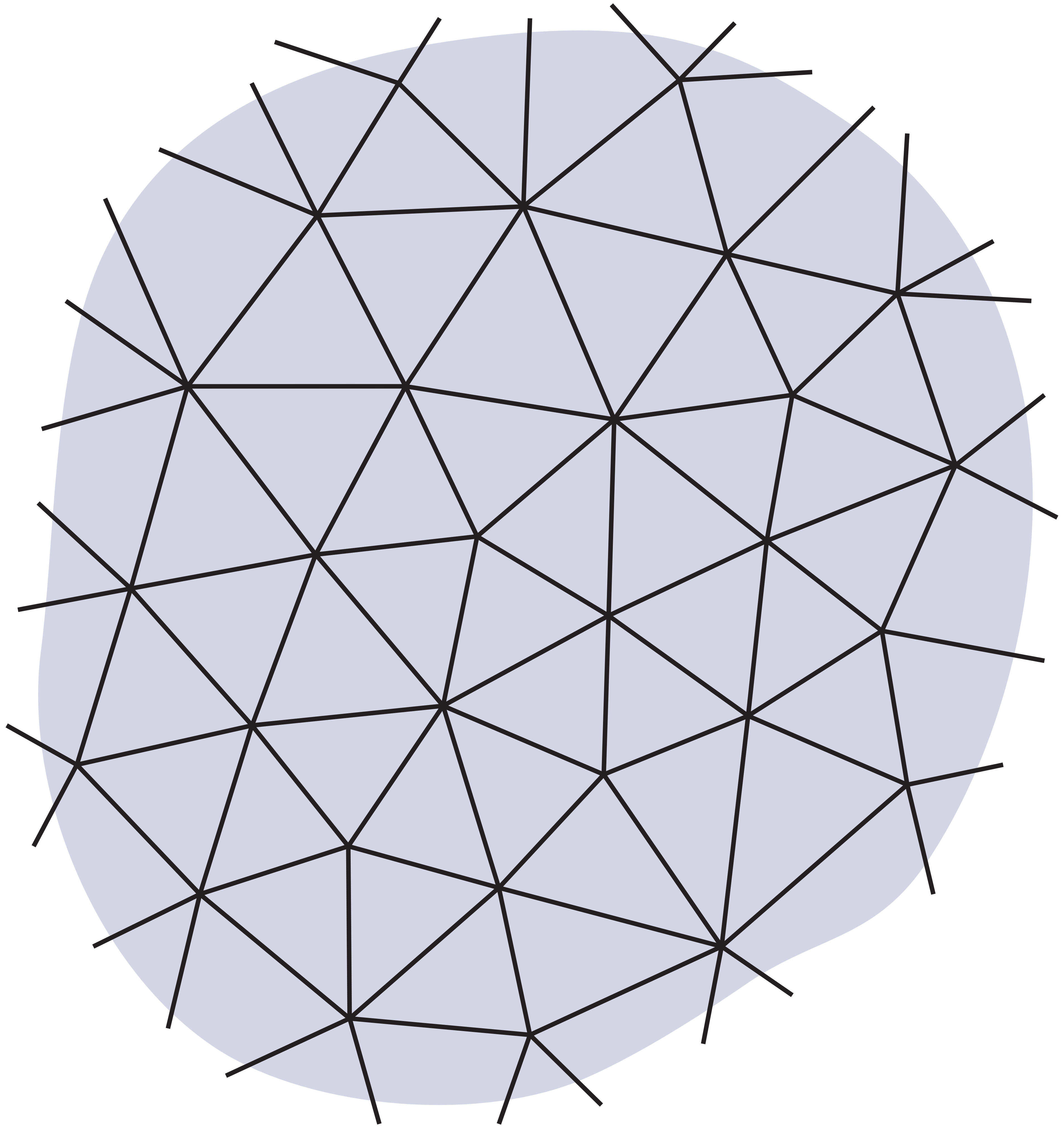
DEC on Triangle Meshes

- Discrete differential forms
  - 0,1,2-forms
- Discrete exterior derivative
  - acting on 0,1-forms



DEC on Triangle Meshes

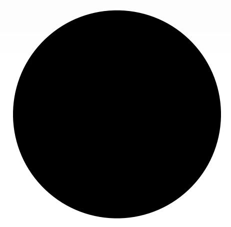
- Discrete differential forms
  - 0,1,2-forms
- Discrete exterior derivative
  - acting on 0,1-forms
- Discrete Hodge-star operator
  - acting on 0,1,2-forms



Discrete differential forms

Discrete differential forms

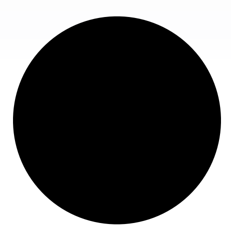
0-form

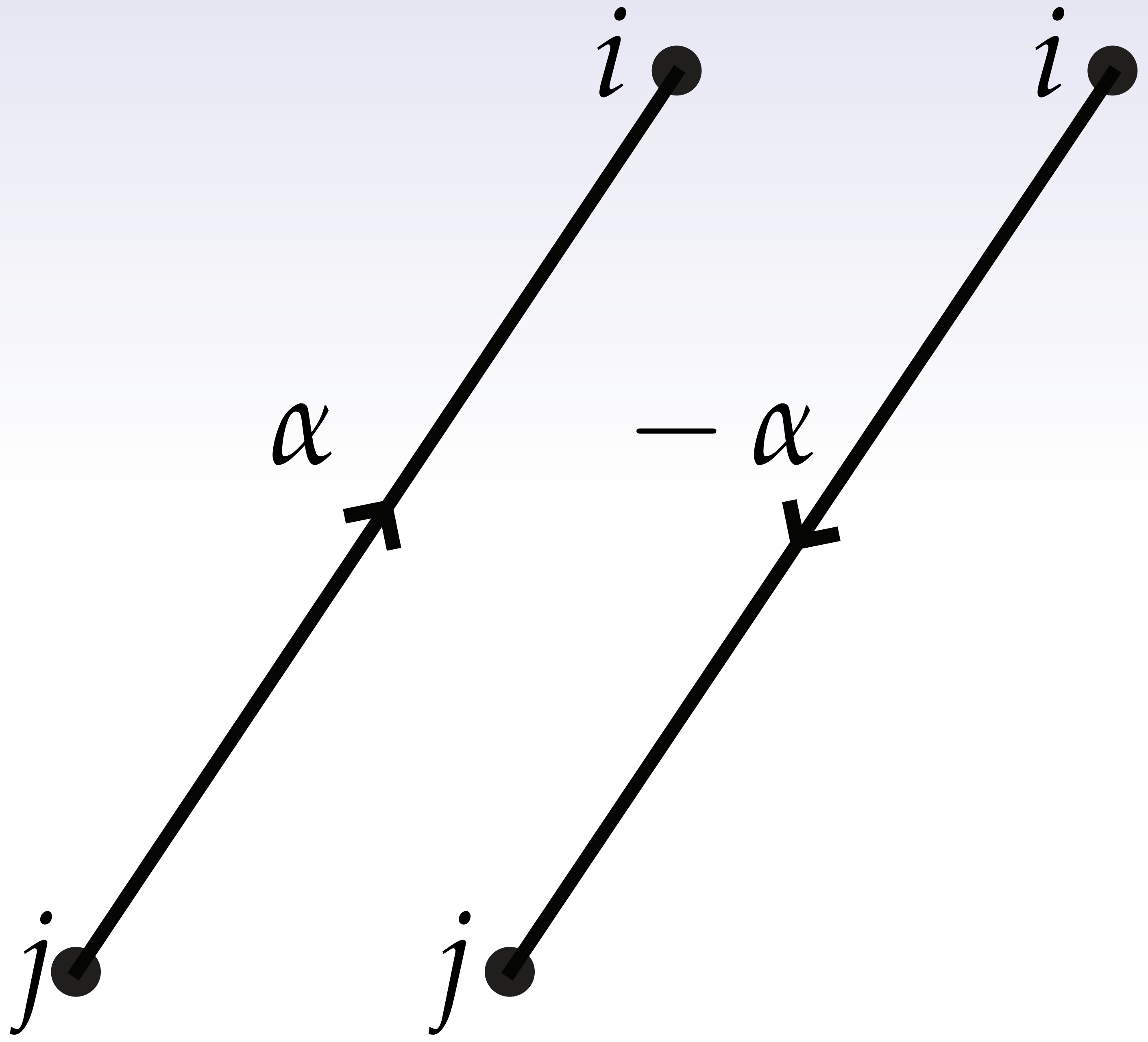


Discrete differential forms

0-form

1-form

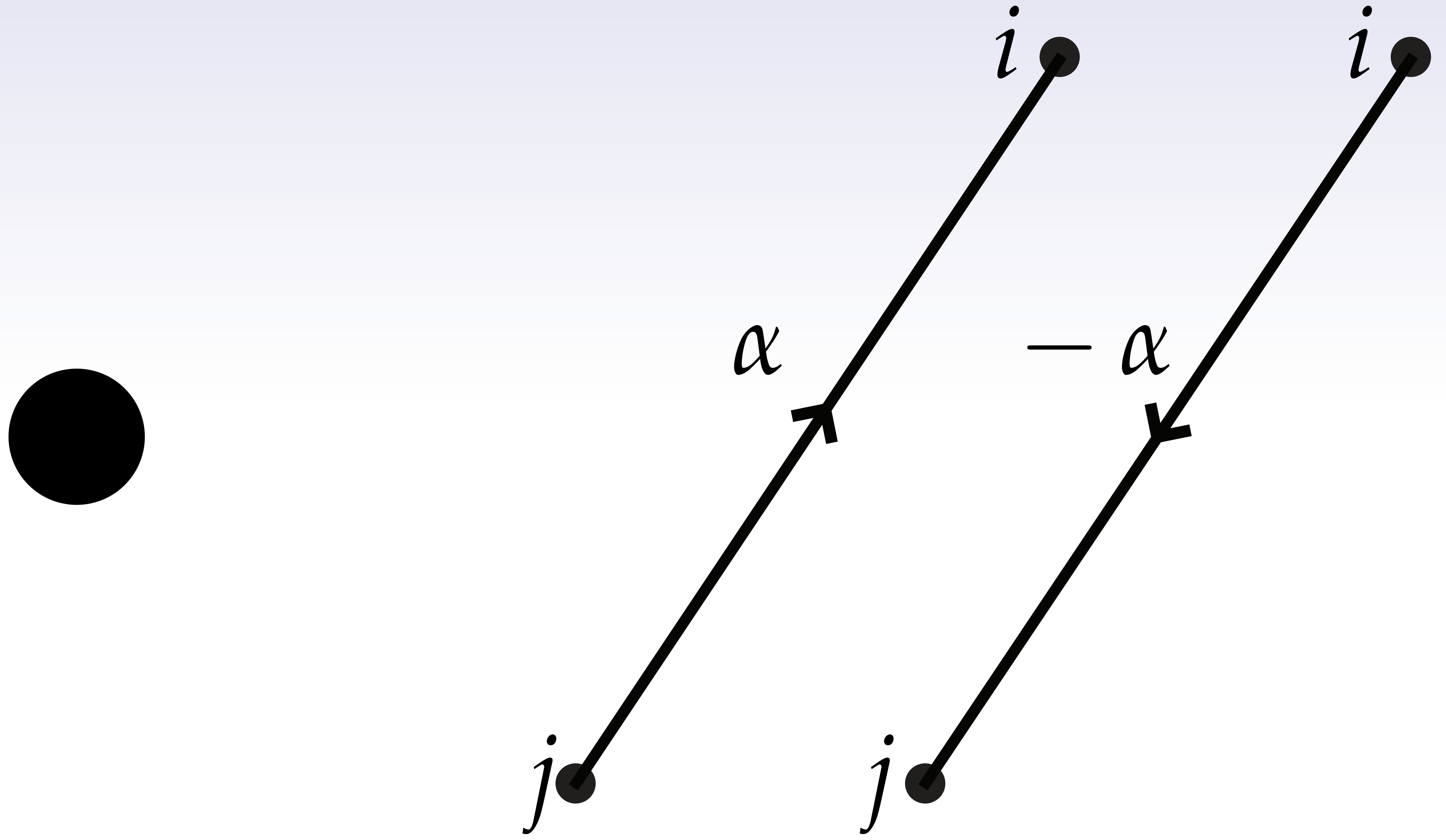




Discrete differential forms

0-form

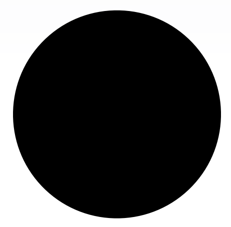
1-form



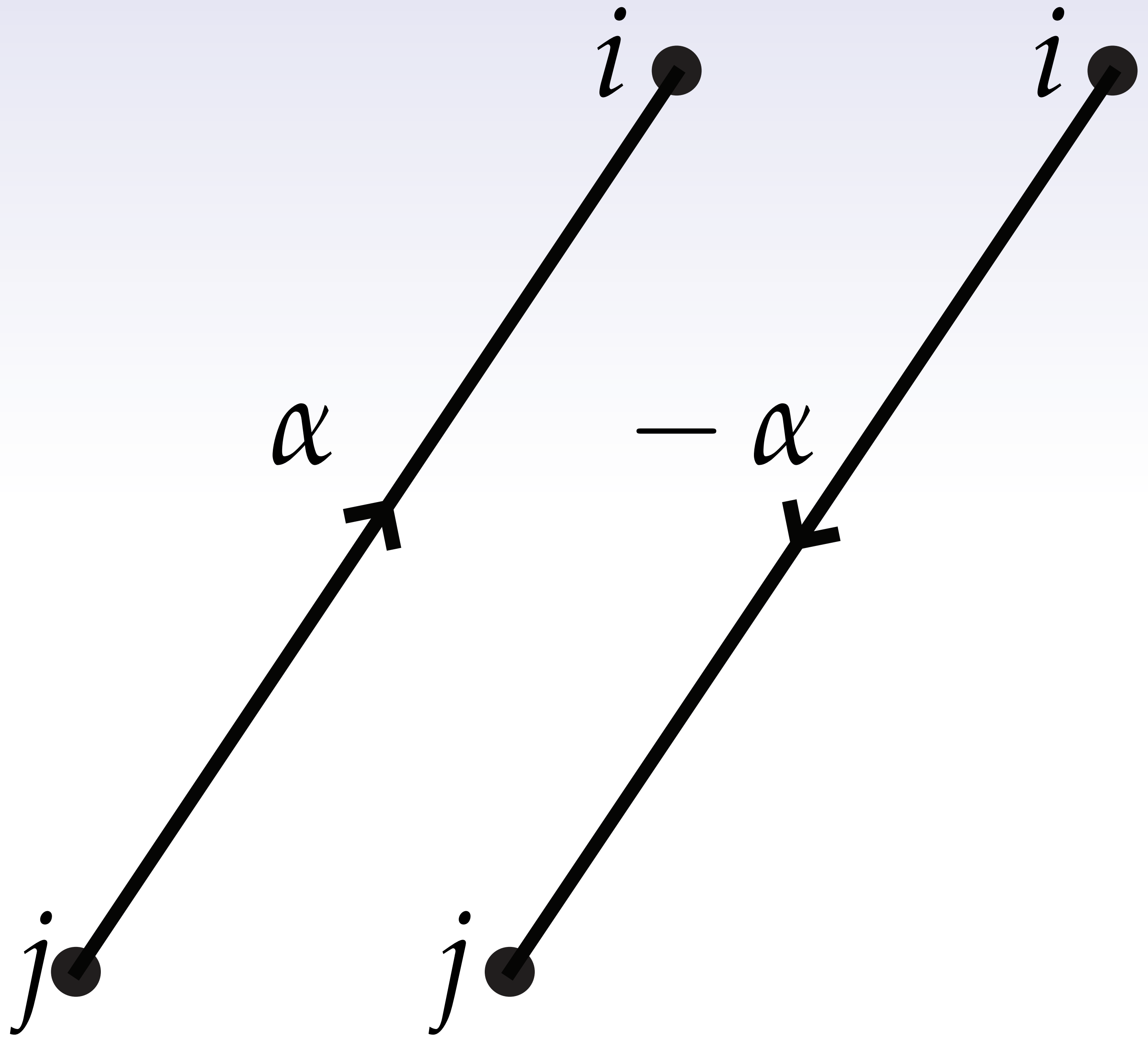
$j < i$

Discrete differential forms

0-form

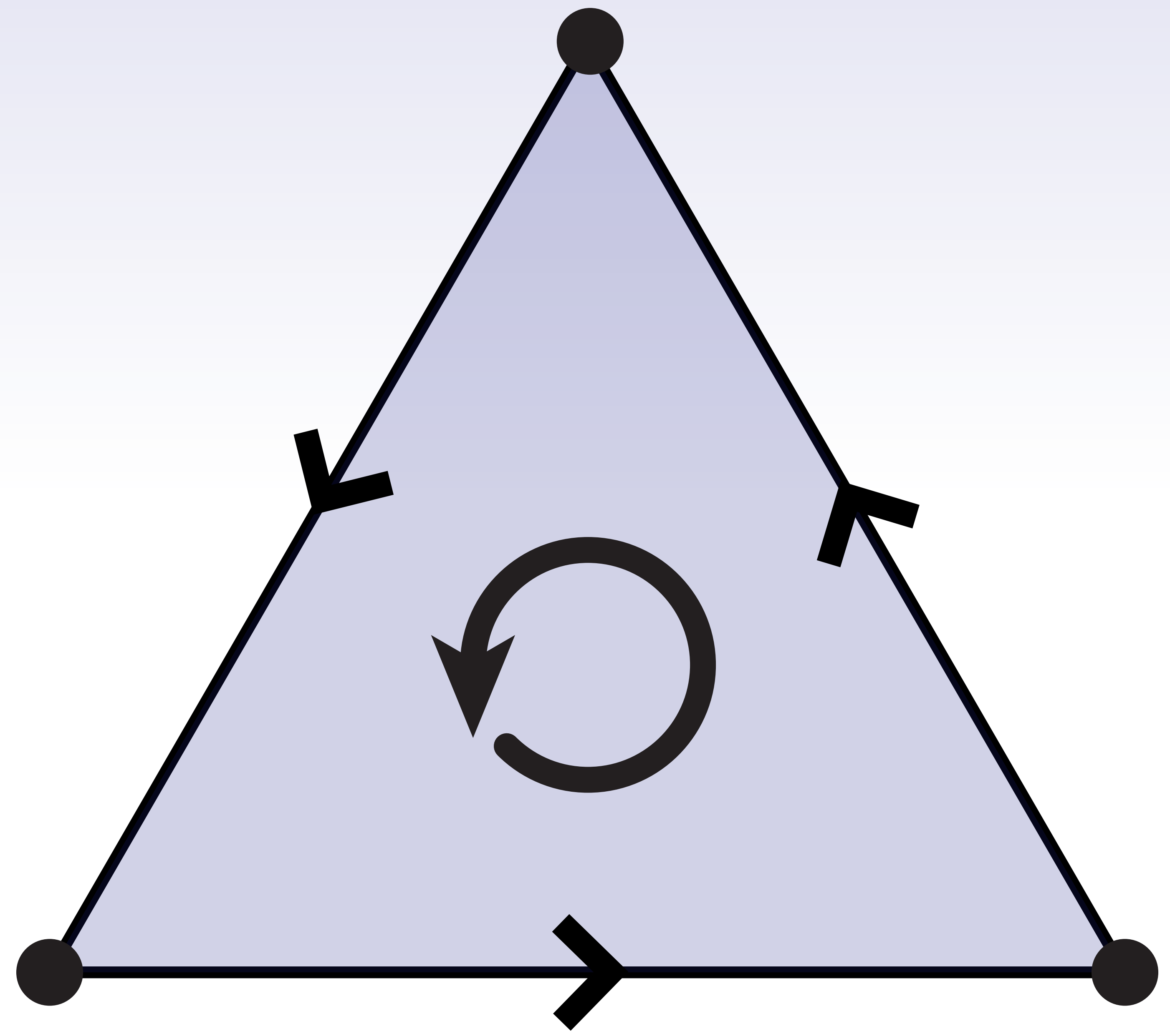


1-form

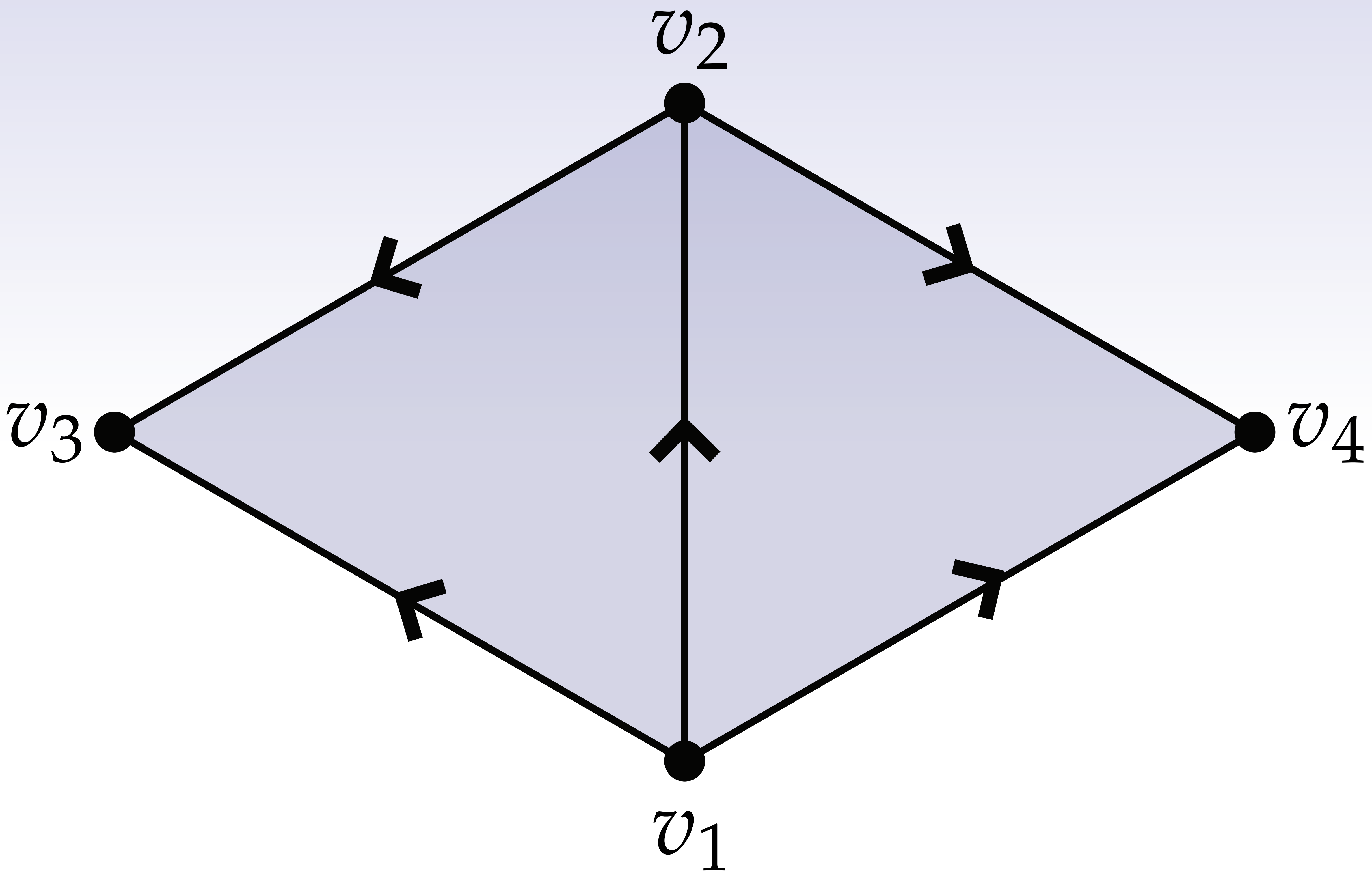


$j < i$

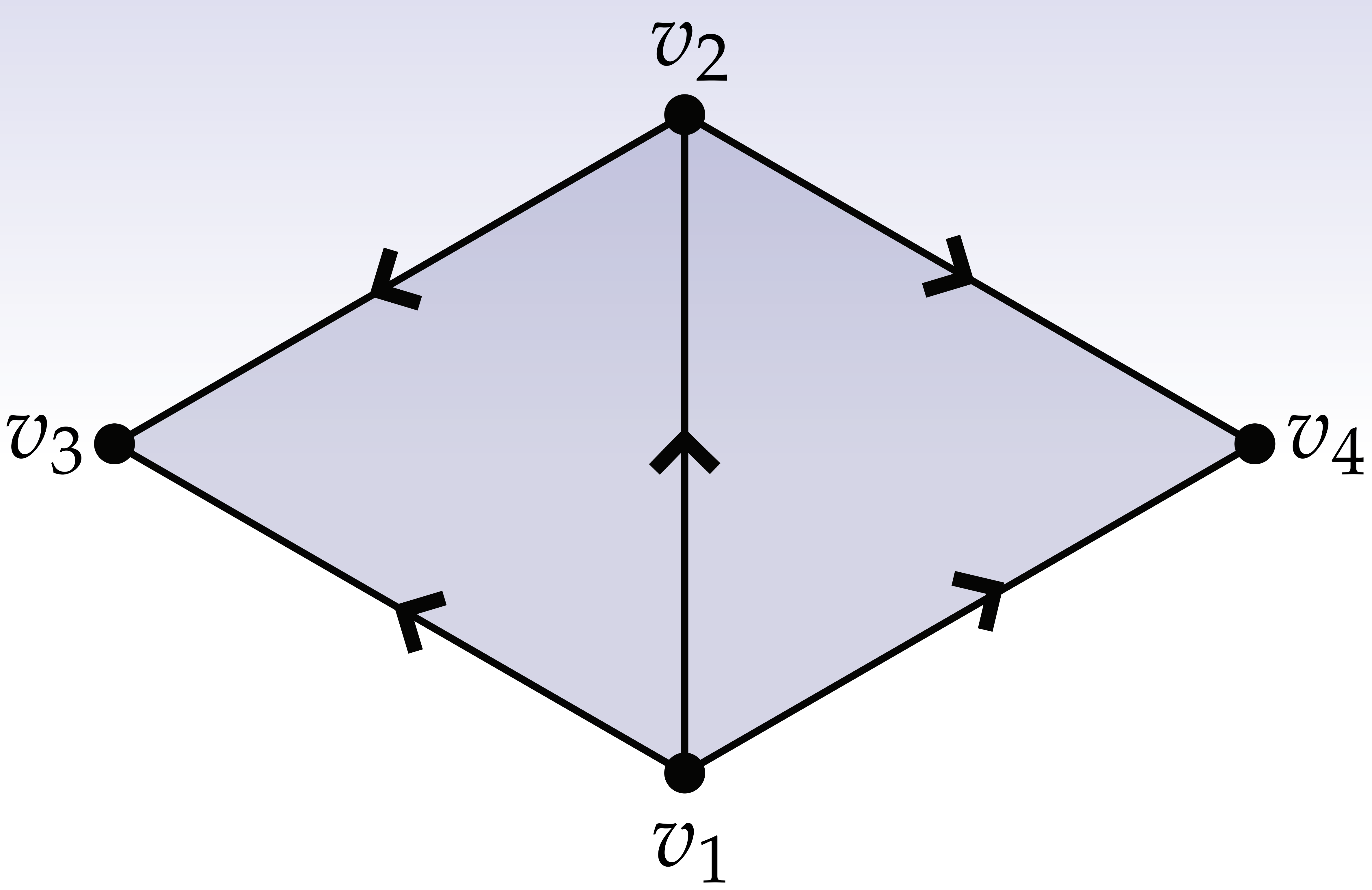
2-form



Discrete exterior derivative for 0-forms

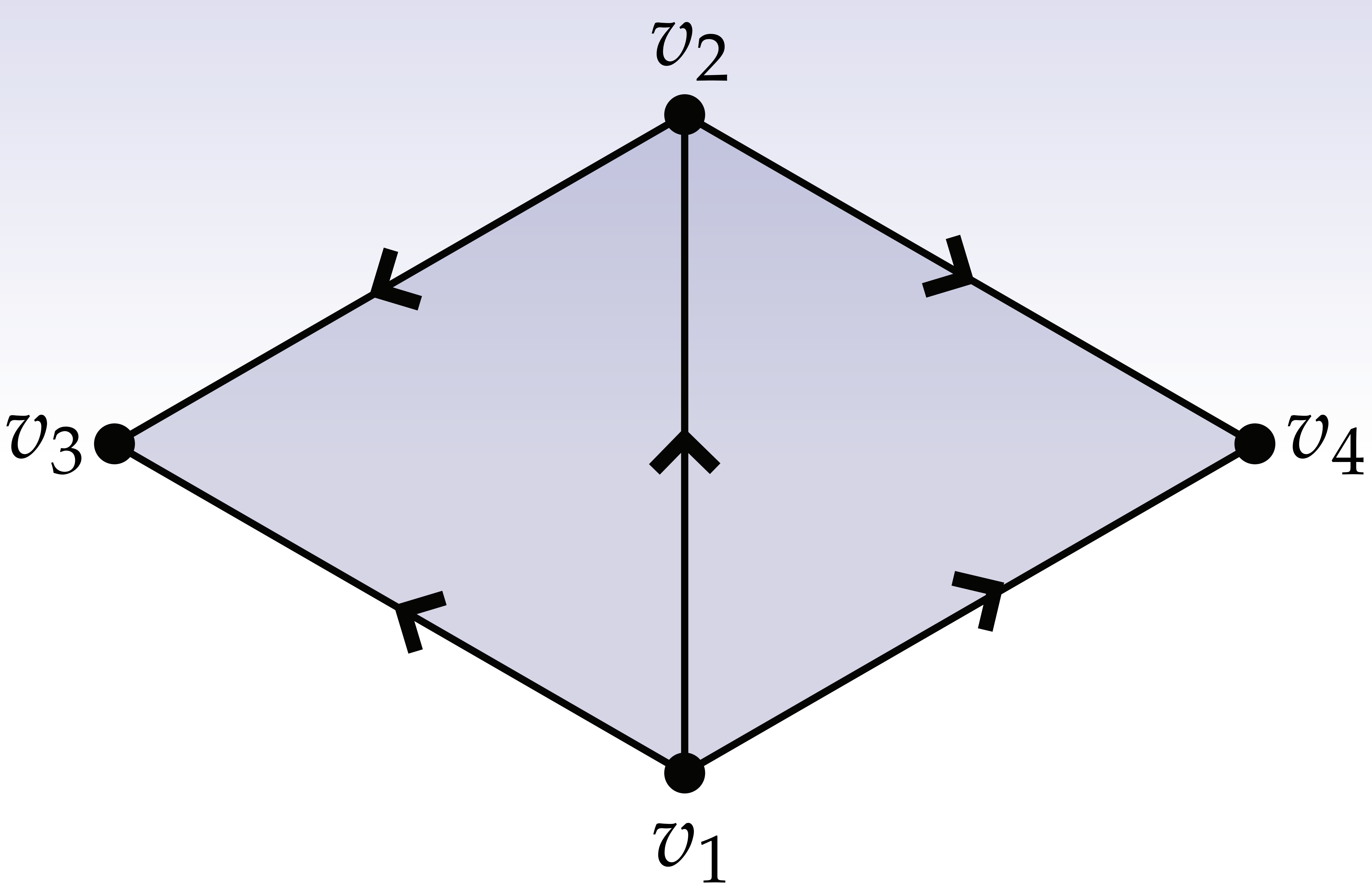


Discrete exterior derivative for 0-forms



$$
\begin{array}{c|cccc}
 & v_1 & v_2 & v_3 & v_4 \\
e_{12} & [\,-1 & +1 & 0 & 0\,] \\
e_{23} & [\,0 & -1 & +1 & 0\,] \\
 & & \cdots & &
\end{array}
$$

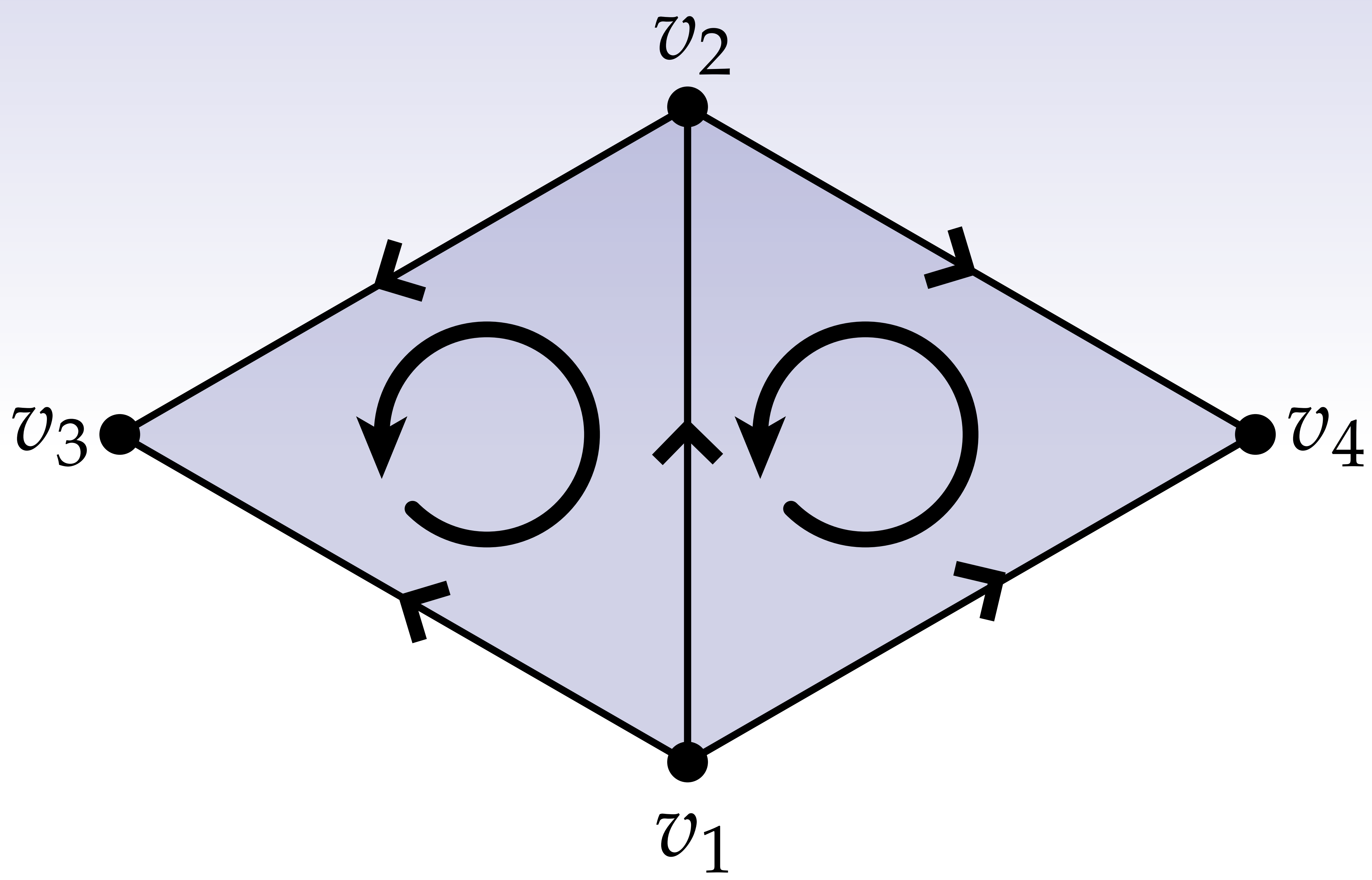
Discrete exterior derivative for 0-forms



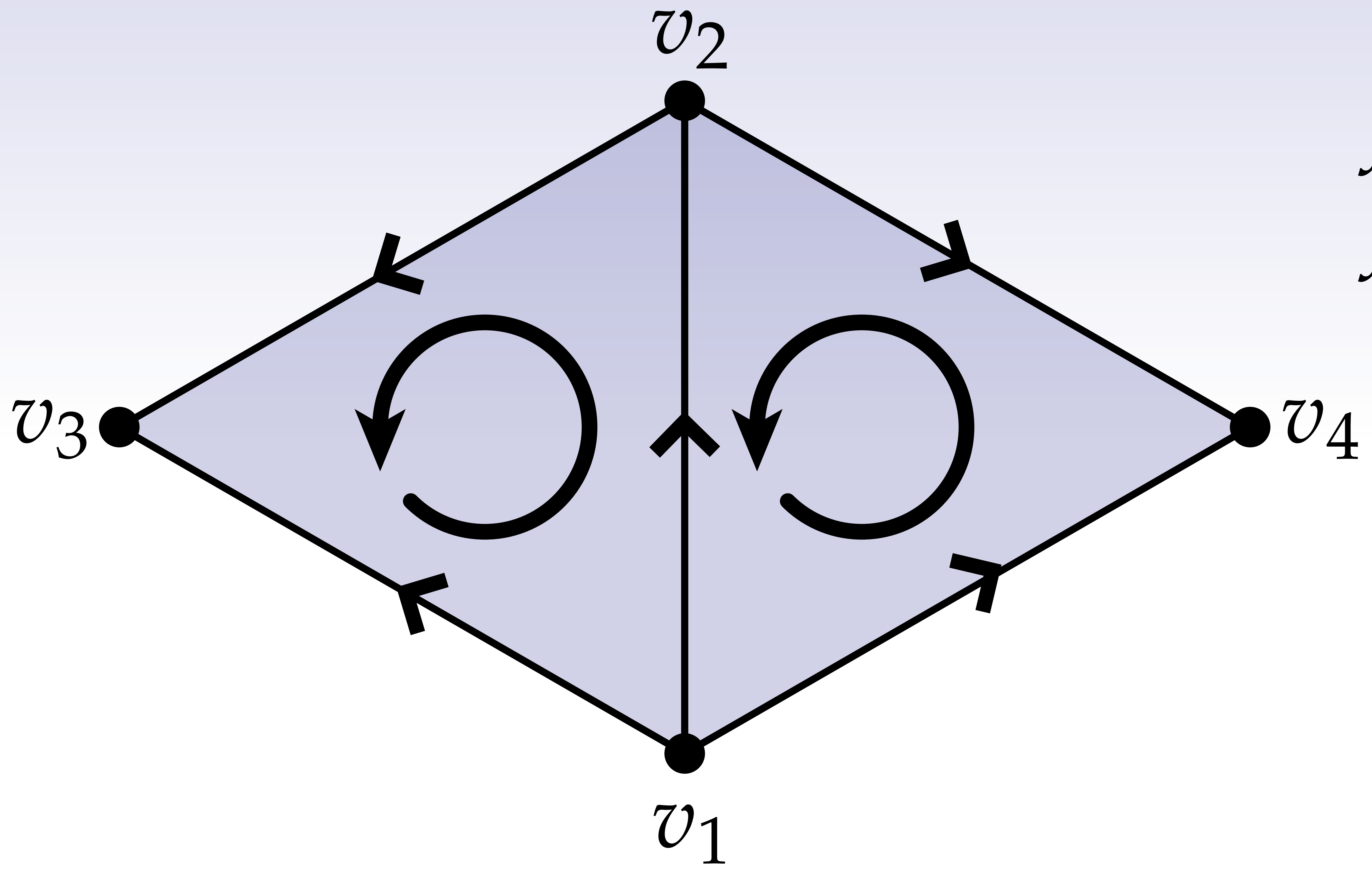
$$
\begin{array}{c c}
 & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
e_{12} & \left[\begin{array}{cccc} -1 & +1 & 0 & 0 \end{array}\right] \\
e_{23} & \left[\begin{array}{cccc} 0 & -1 & +1 & 0 \end{array}\right] \\
 & \cdots
\end{array}
$$

```
SparseMatrix<Type> d0(#Edges, #Vertices);
foreach edge eij:
   d0(eij, vi) = (i < j)? -1 : -1;
   d0(eij, vj) = (i < j)? +1 : -1;
```

Discrete exterior derivative for 1-forms

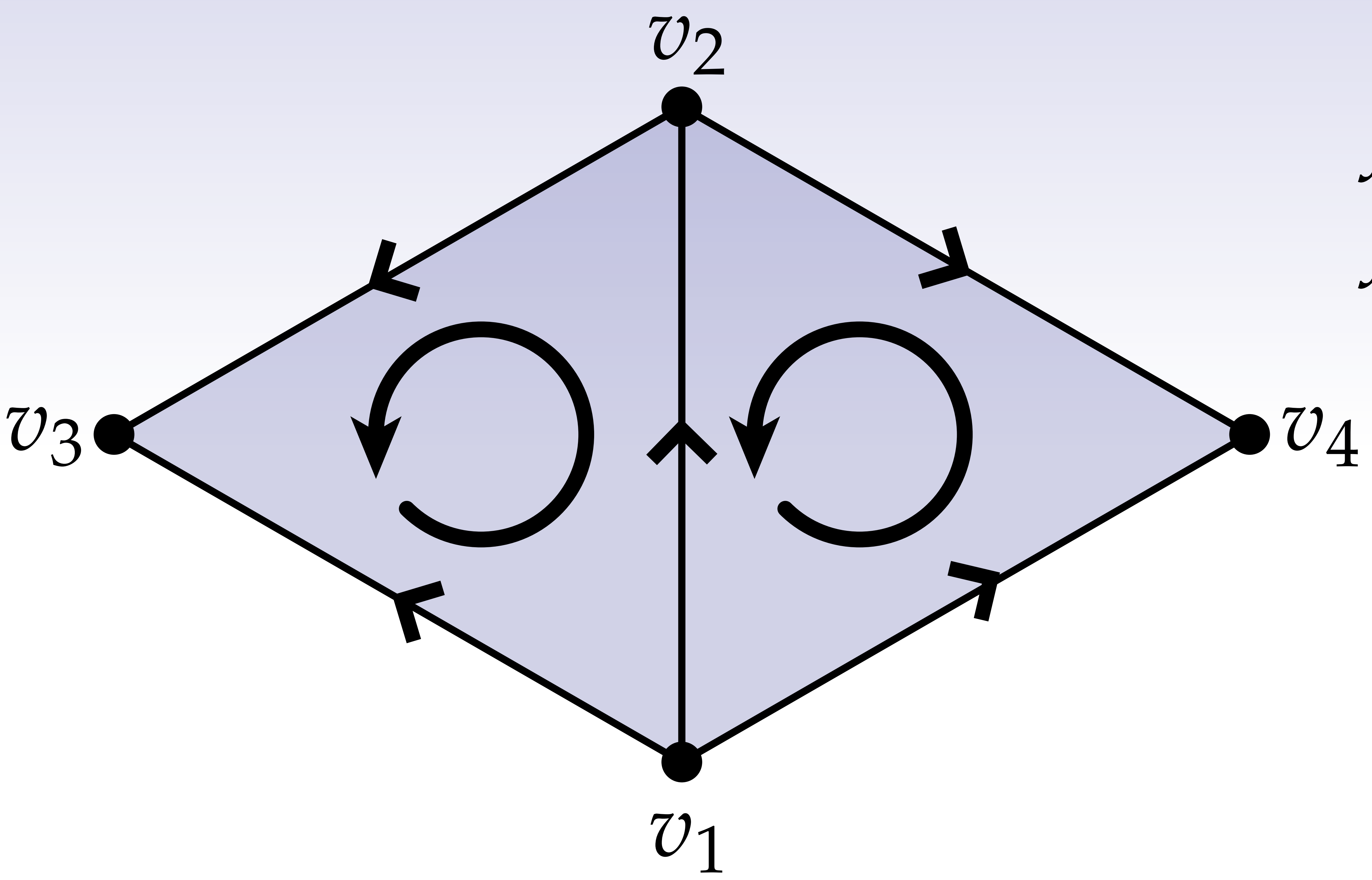


Discrete exterior derivative for 1-forms



$$\begin{array}{cc} & \begin{array}{ccccc} e_{12} & e_{23} & e_{13} & e_{14} & e_{24} \end{array} \\ f_{123} & [\begin{array}{ccccc} +1 & +1 & -1 & 0 & 0 \end{array}] \\ f_{214} & [\begin{array}{ccccc} -1 & 0 & 0 & +1 & -1 \end{array}] \end{array}$$

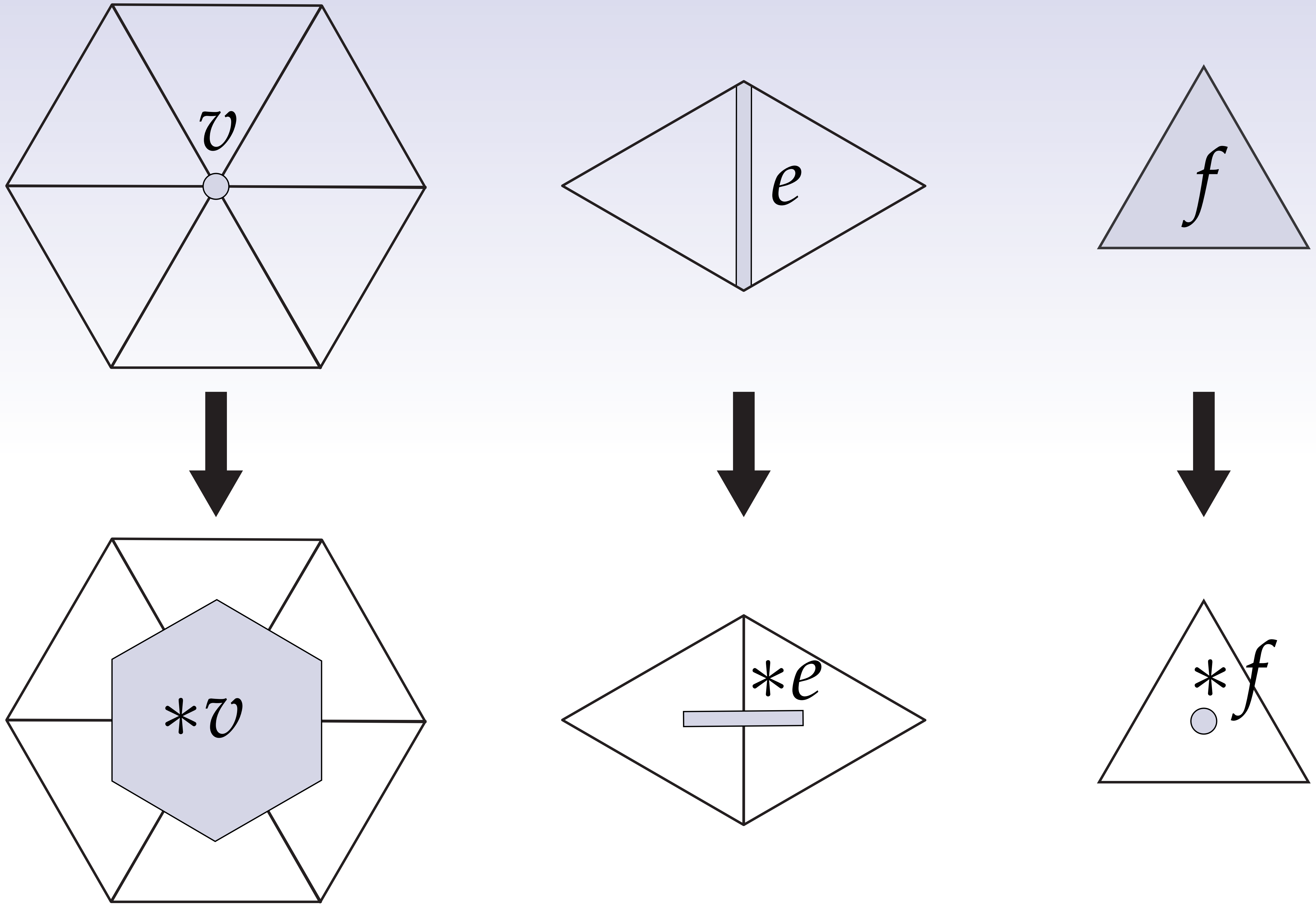
Discrete exterior derivative for 1-forms



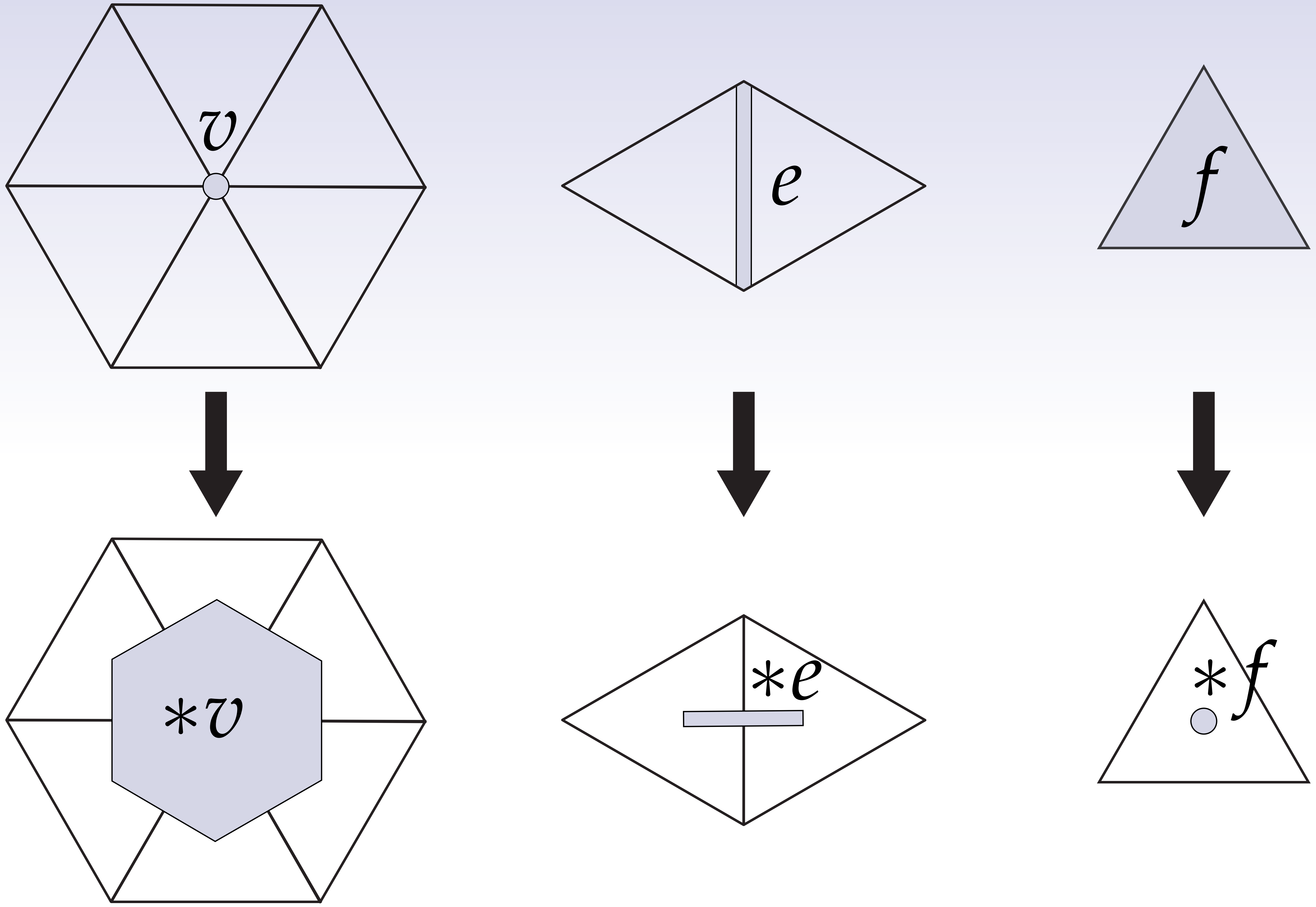
$$\begin{array}{c c}
 & \begin{array}{ccccc} e_{12} & e_{23} & e_{13} & e_{14} & e_{24} \end{array} \\
f_{123} & \begin{bmatrix} +1 & +1 & -1 & 0 & 0 \end{bmatrix} \\
f_{214} & \begin{bmatrix} -1 & 0 & 0 & +1 & -1 \end{bmatrix}
\end{array}$$

```
SparseMatrix<Type> d1(#Faces,#Edges);

foreach face fijk:

   d1(fijk, eij) = (i < j)? +1 : −1;

   d1(fijk, ejk) = (j < k)? +1 : −1;

   d1(fijk, eki) = (k < i)? +1 : −1;
```

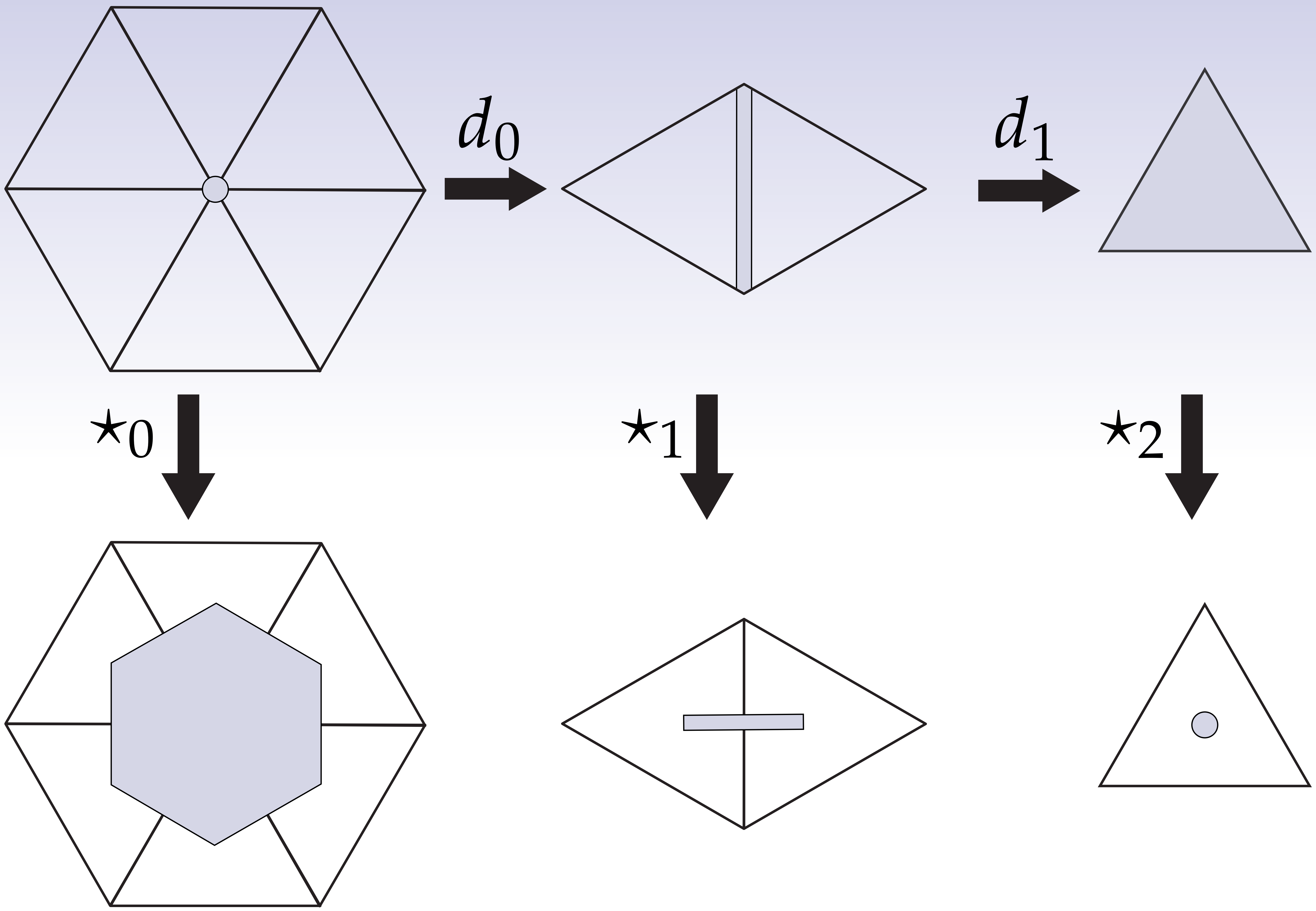
Discrete Hodge-star

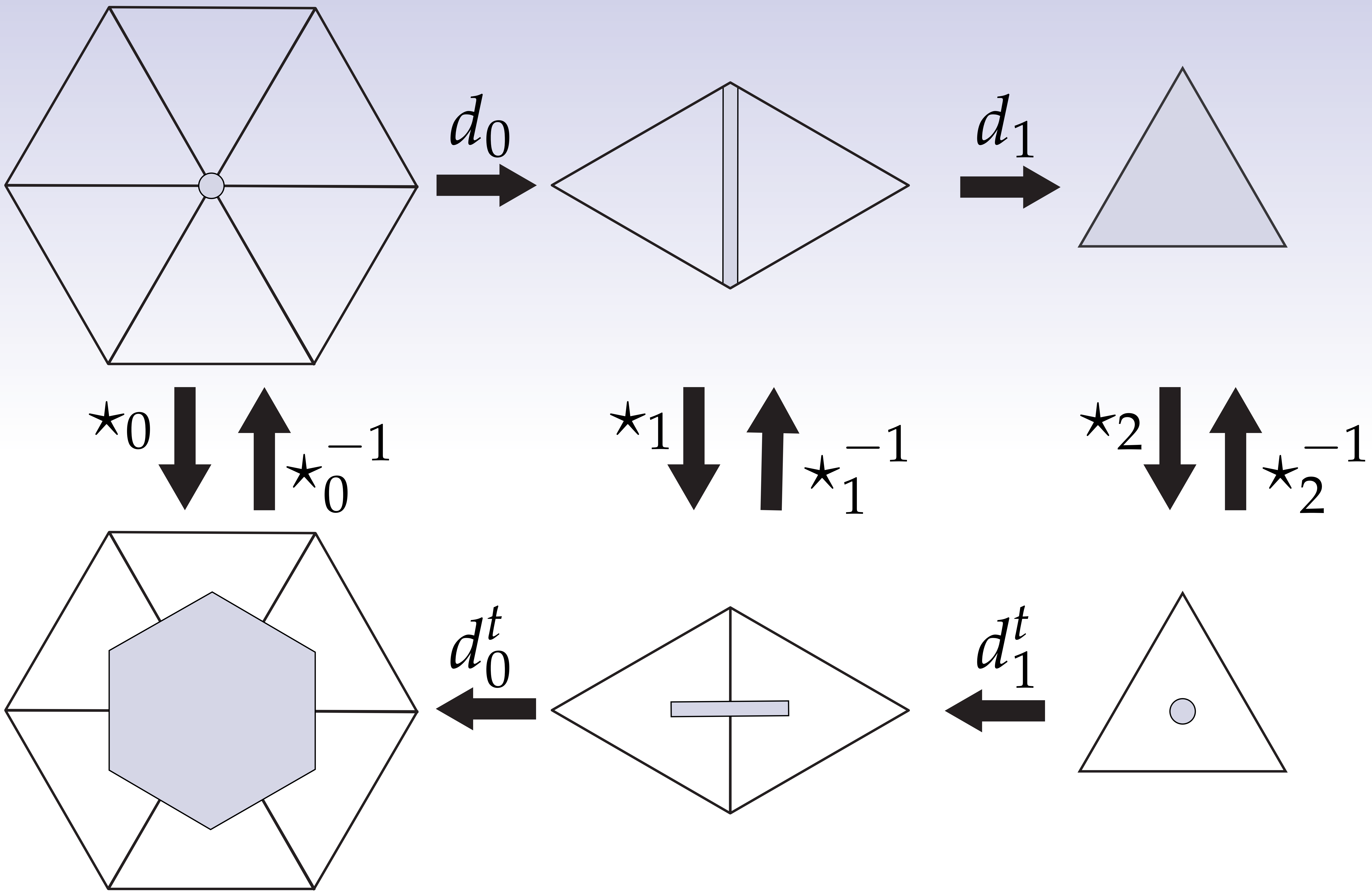


Discrete Hodge-star



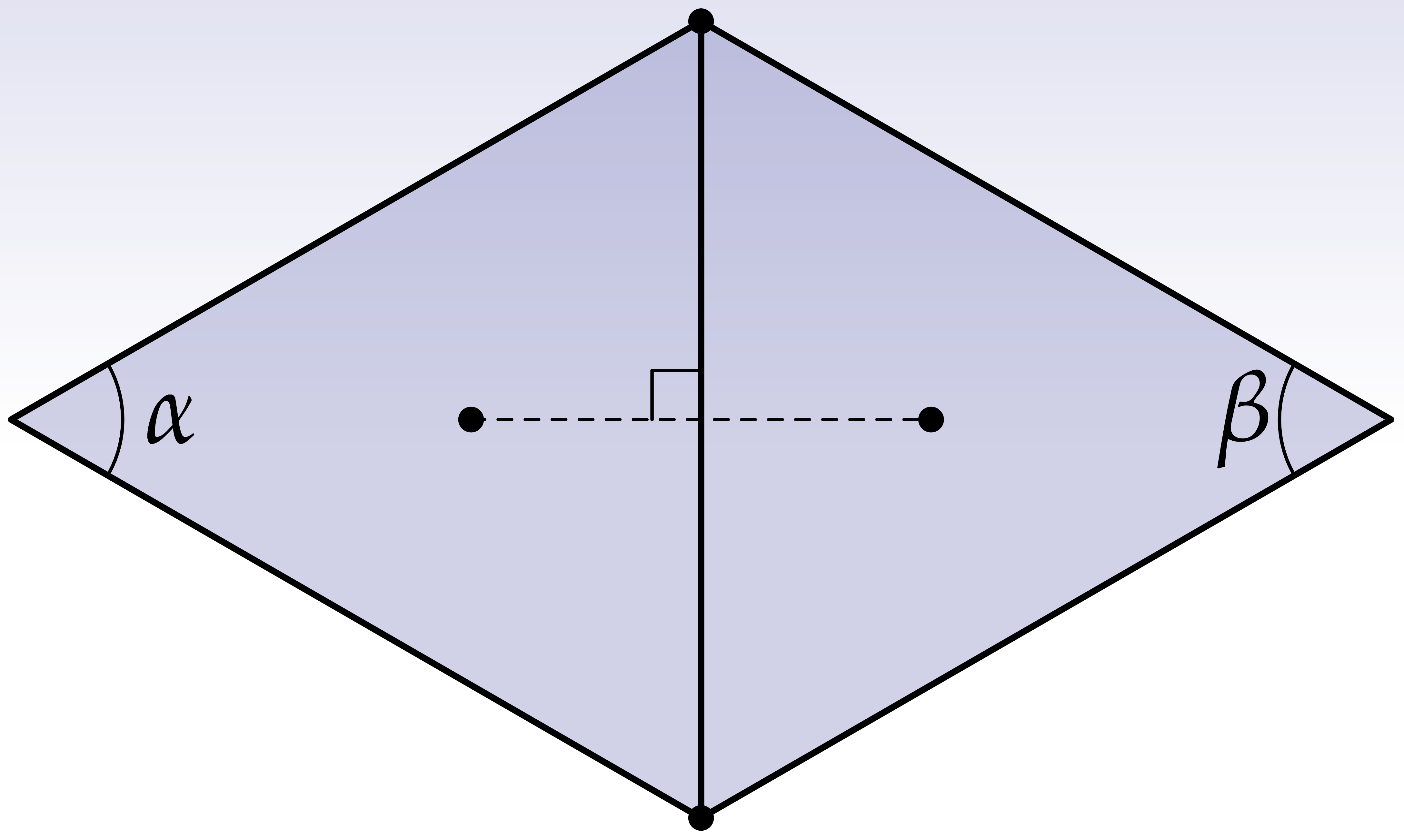

```
SparseMatrix<Type> star0;
foreach vertex v:
   star0(v,v) = area(v);

SparseMatrix<Type> star1;
foreach edge e:
   star1(e,e) = len(*e) / len(e)

SparseMatrix<Type> star2;
foreach face f:
   star2(f,f) = 1 / area(f);
```


d_0
d_1
$\star_0$
$\star_1$
$\star_2$



d_0
d_1
$\star_0$
$\star_0^{-1}$
$\star_1$
$\star_1^{-1}$
$\star_2$
$\star_2^{-1}$
d_0^t
d_1^t

Circumcentric dual




```
SparseMatrix<Type> star1;
foreach edge e:
   star1(e,e) = 0.5*(cot(alpha) + cot(beta));
```

$$\star_e = \frac{l_{*e}}{l_e} = \frac{1}{2}\left(\cot\alpha + \cot\beta\right)$$

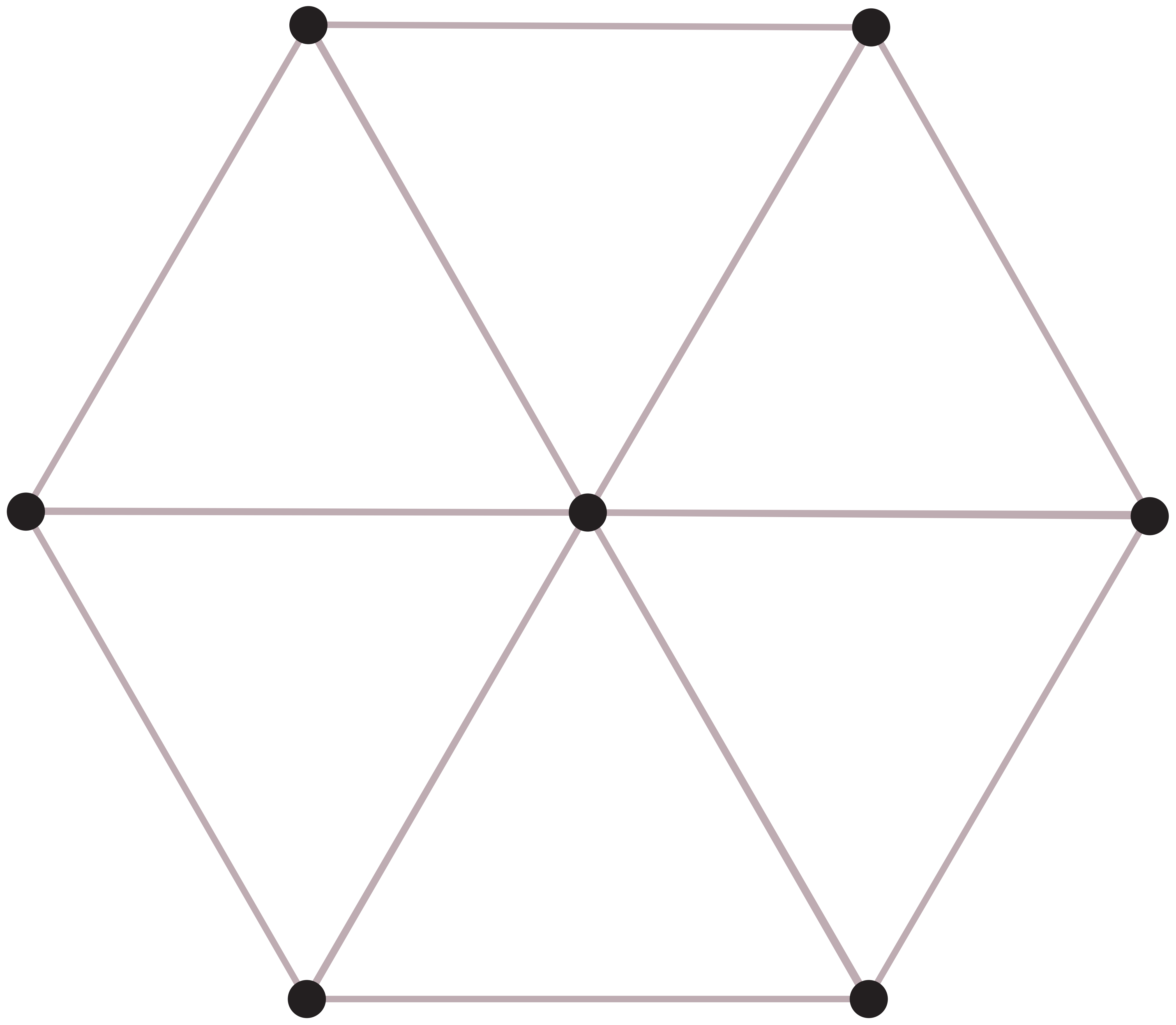
Discrete Laplacian

Discrete Laplacian

```
SparseMatrix<Real> L = d0^t * star1 * d0;
```

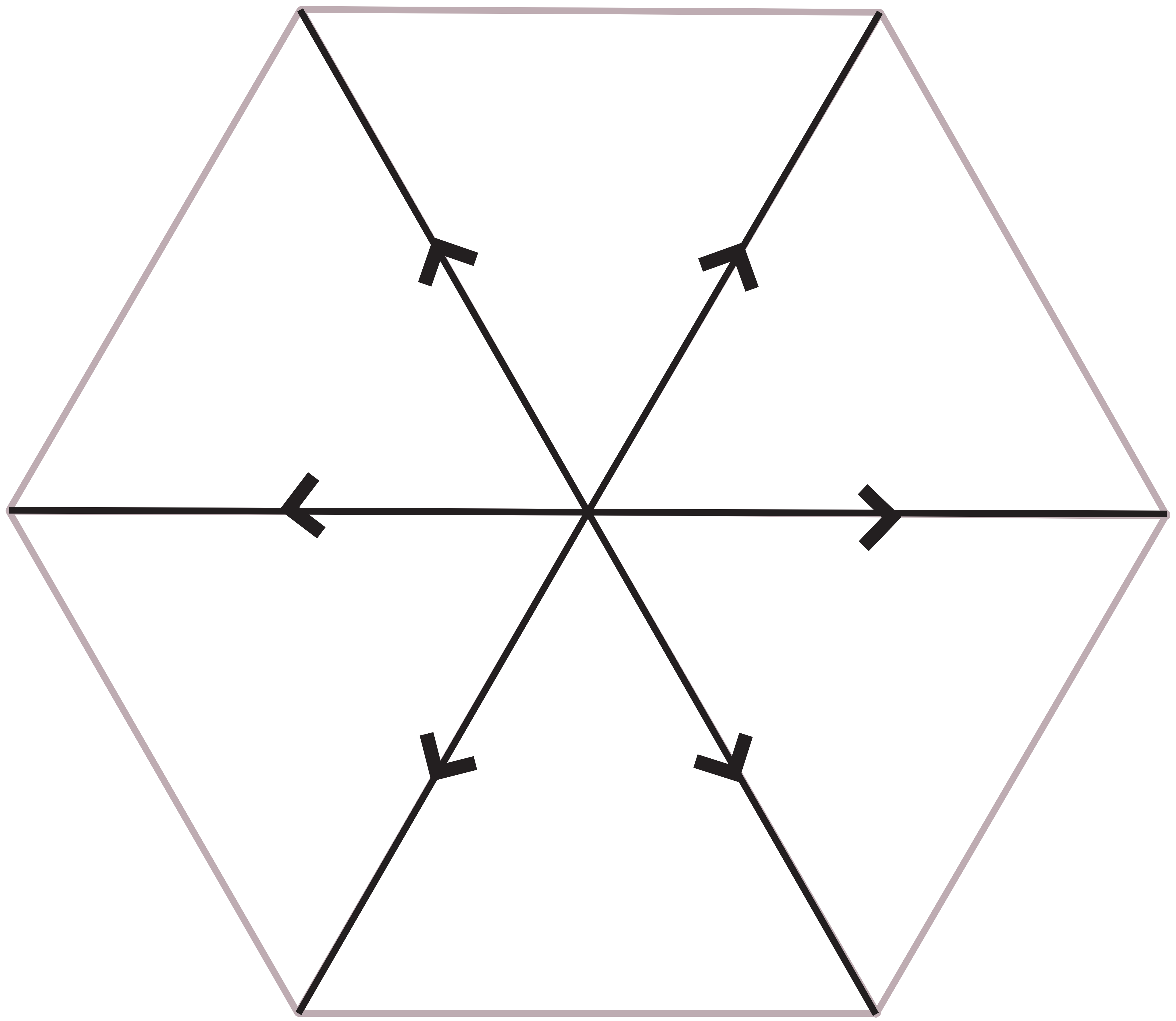
Discrete Laplacian

```
SparseMatrix<Real> L = d0^t * star1 * d0;
```



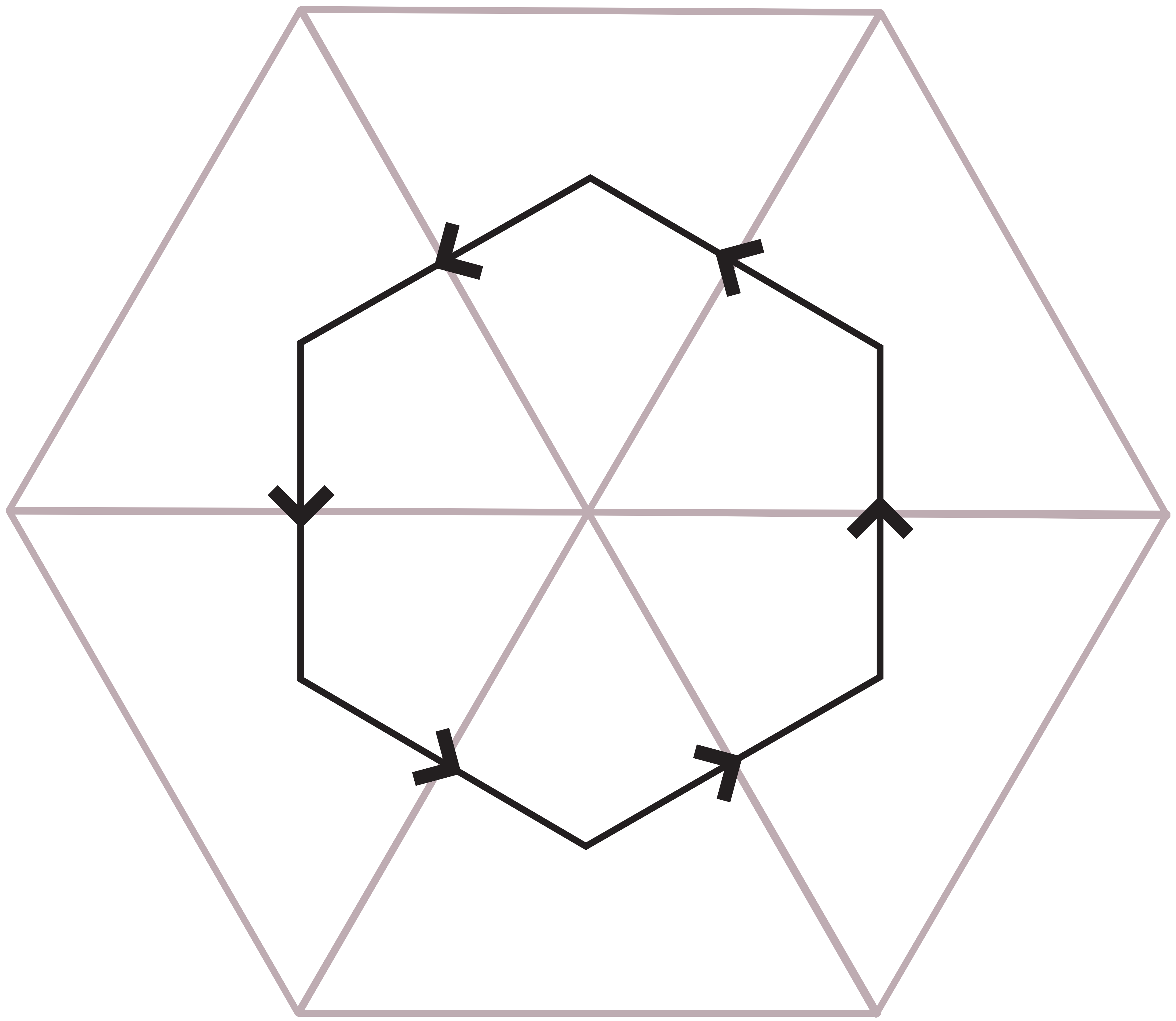
Discrete Laplacian

```
SparseMatrix<Real> L = d0^t * star1 * d0;
```



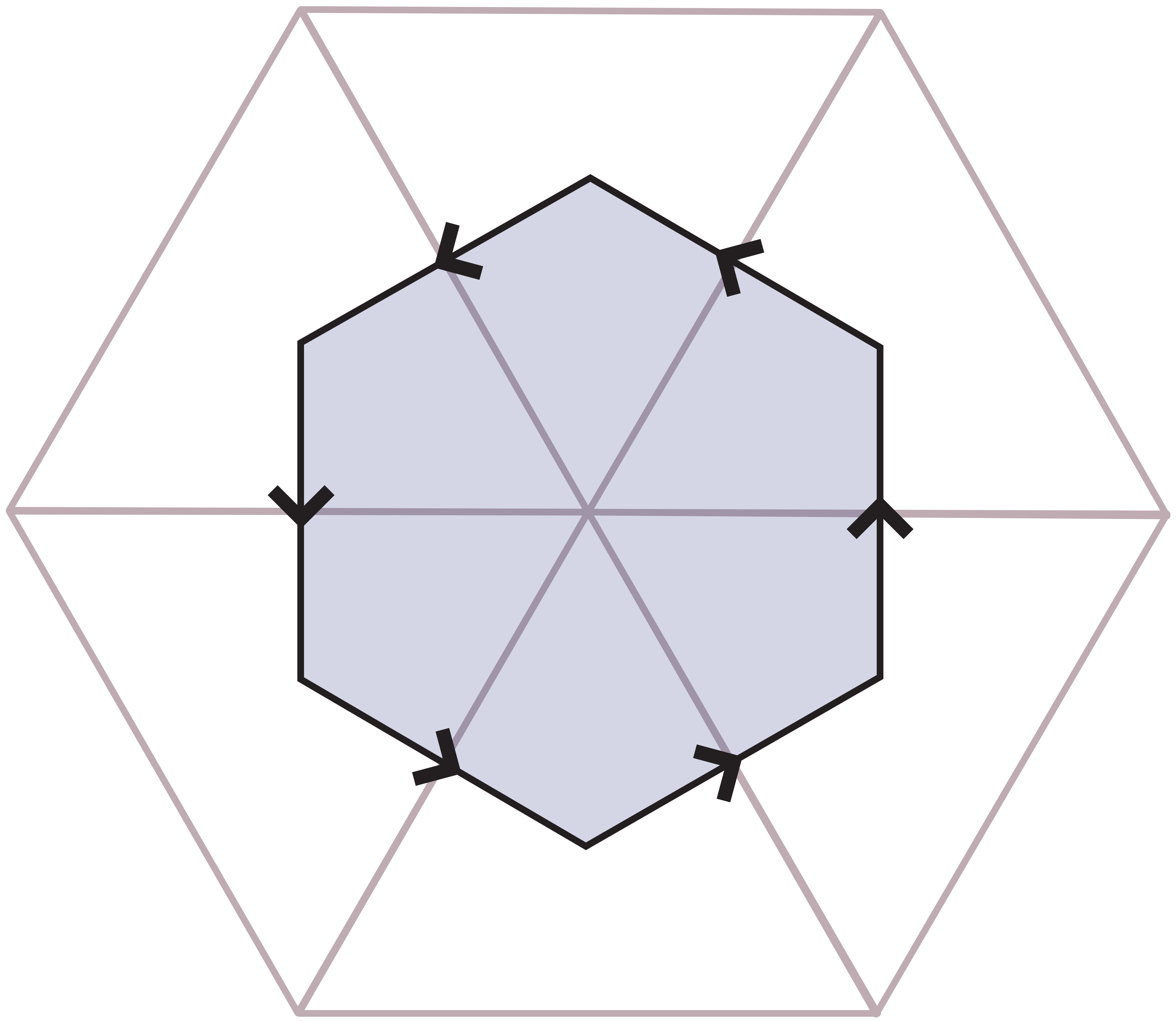
Discrete Laplacian

```
SparseMatrix<Real> L = d0^t * star1 * d0;
```

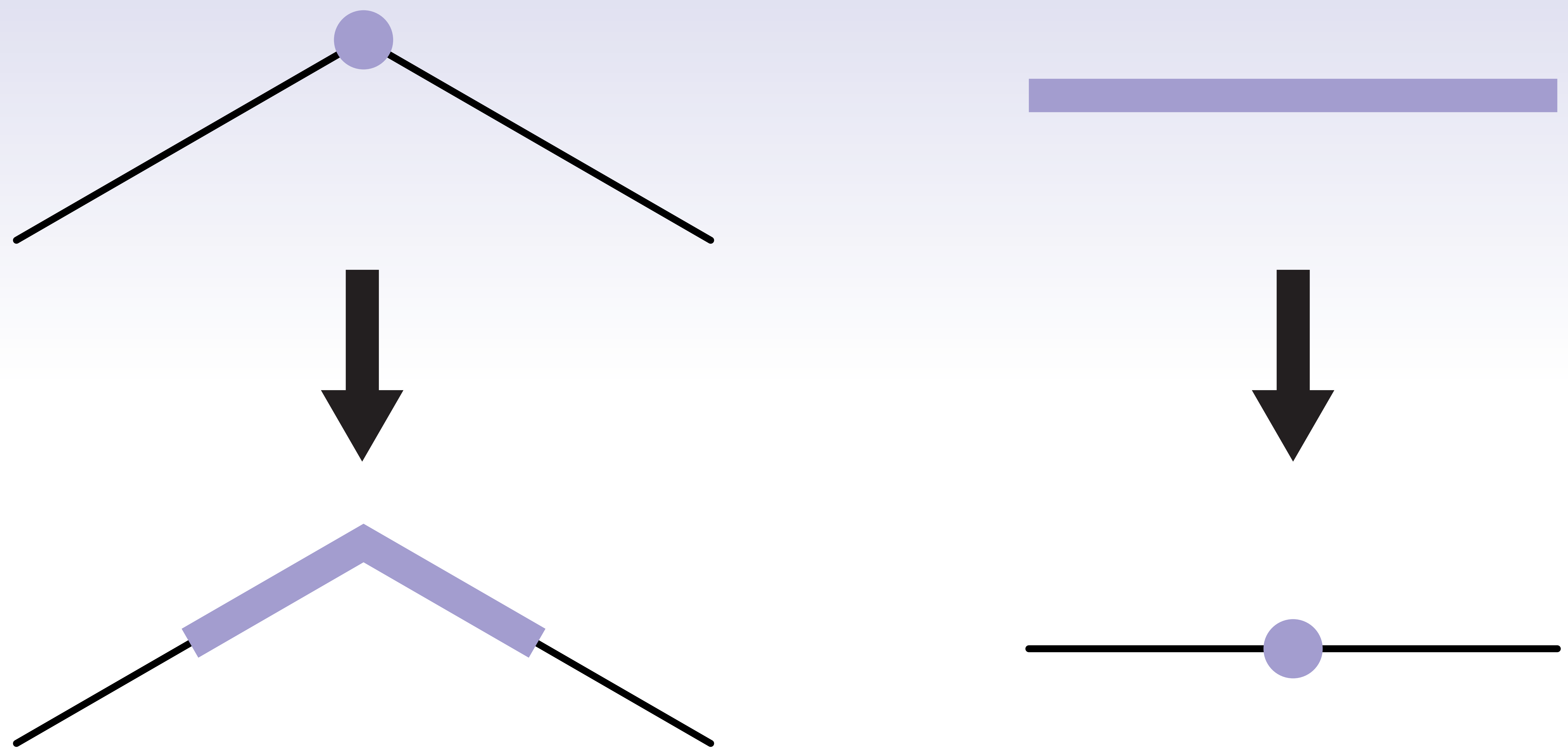


Discrete Laplacian

```
SparseMatrix<Real> L = d0^t * star1 * d0;
```



DEC on polygonal curves

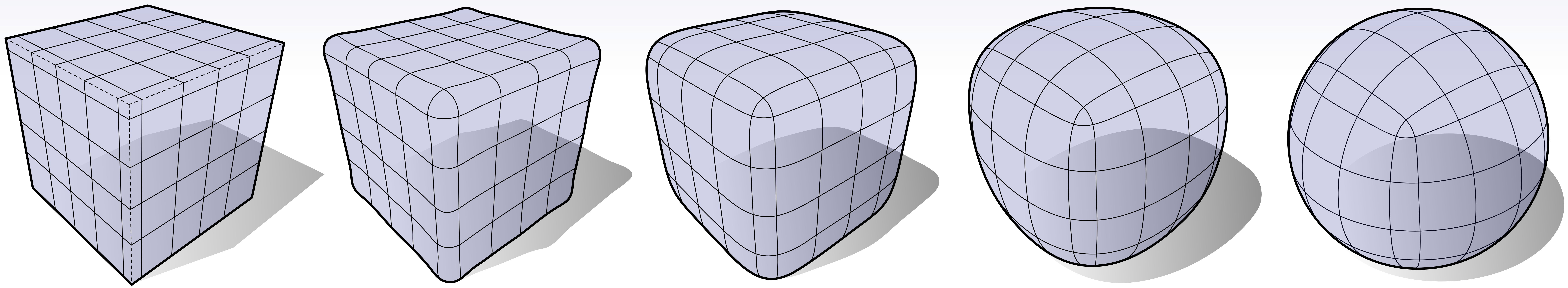


DEC on tetrahedral meshes

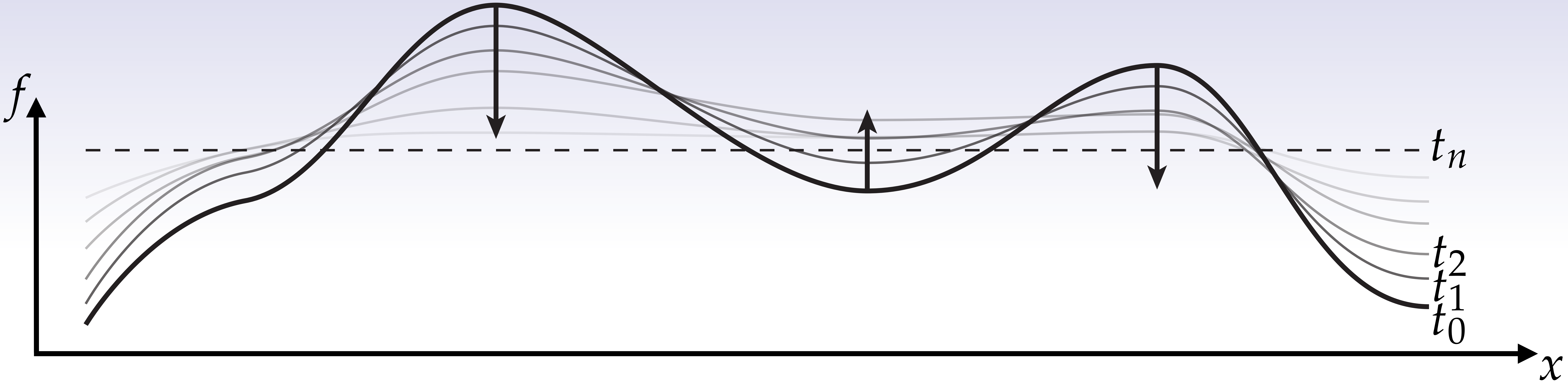
Smoothing

Problem Statement

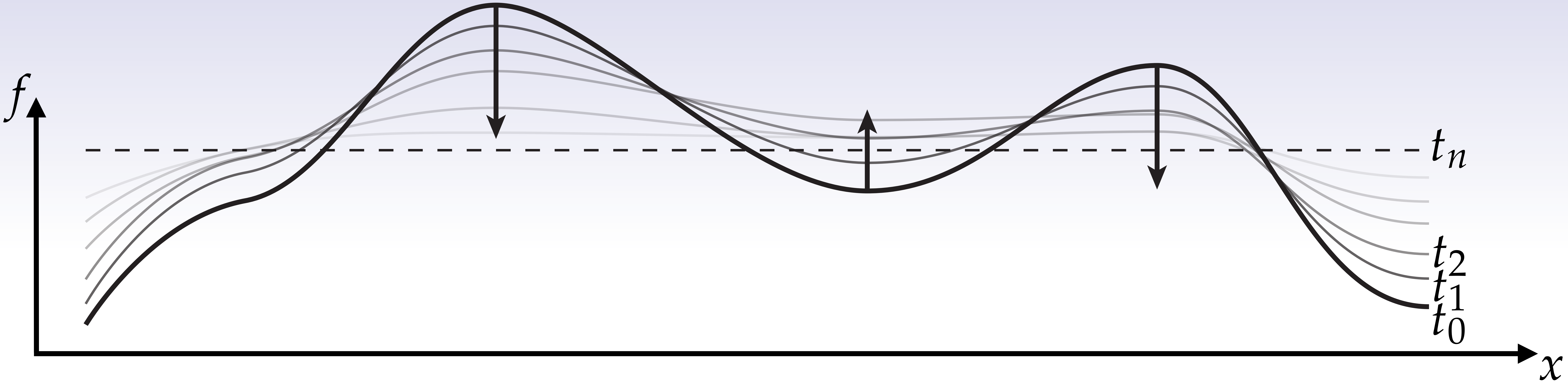
course notes: chapter 6.6
code/Fairing



Motivated by diffusion

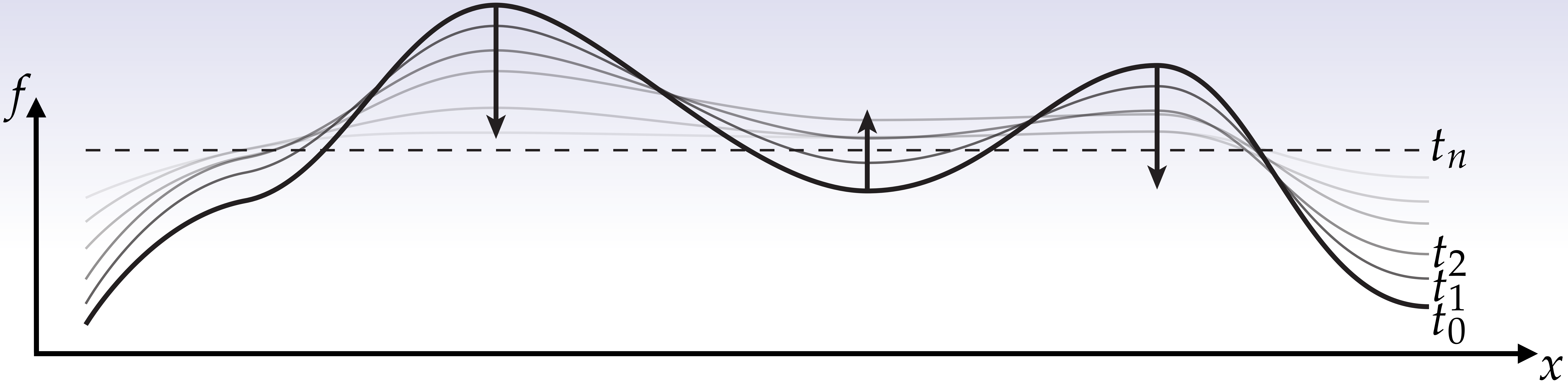


Motivated by diffusion



$$\partial_t f = \partial^2_{x,x} f$$

Motivated by diffusion



$$\partial_t f = \partial^2_{x,x} f$$

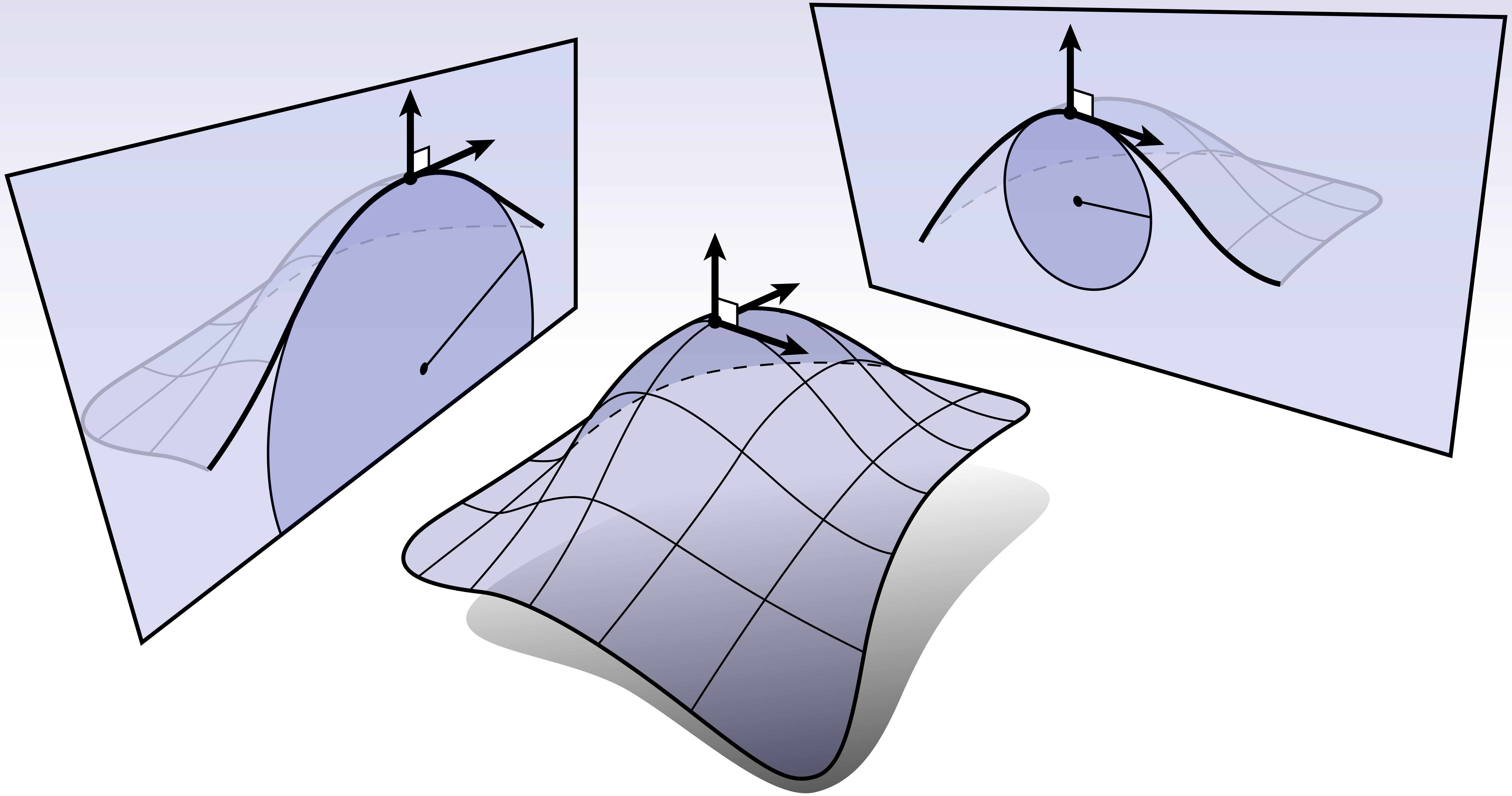
$$\partial_t f = \Delta f$$

Mean Curvature flow

$$\partial_t f = \Delta f$$

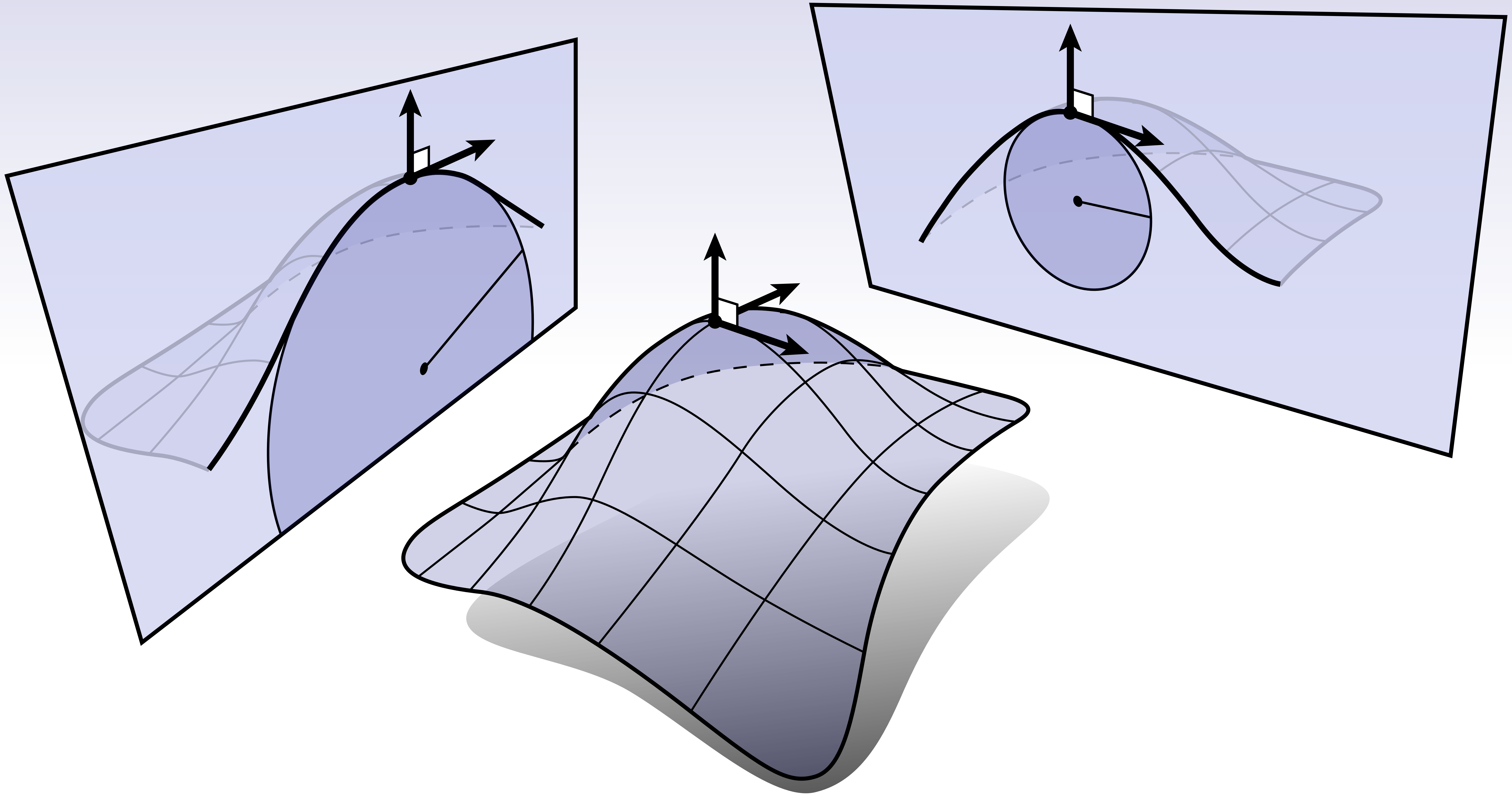
Mean Curvature flow

$$\partial_t f = \Delta f$$



Mean Curvature flow

$$\partial_t f = \Delta f$$



$$\Delta f = 2HN$$

Discretization

$$\partial_t f = \Delta f$$

Discretization

$$\partial_t f = \Delta f$$

$$\Big\downarrow \text{semi-implicit}$$

Desbrun et al. 1999

Discretization

$$\partial_t f = \Delta f$$

$$\Big\downarrow \text{semi-implicit}$$

$$\frac{f_h - f_0}{h} = \Delta_0 f_h$$

Desbrun et al. 1999

Discretization

$$\partial_t f = \Delta f$$

$$\Big\downarrow \text{semi-implicit}$$

$$\frac{f_h - f_0}{h} = \Delta_0 f_h$$

$$\Big\downarrow$$

$$(I - h\Delta_0)\, f_h = f_0$$

Desbrun et al. 1999

Discretization

$$\partial_t f = \Delta f$$

$$\Big\downarrow \text{semi-implicit}$$

$$\frac{f_h - f_0}{h} = \Delta_0 f_h$$

$$\Big\downarrow$$

$$\left(I - h\Delta_0\right) f_h = f_0$$

$$\Big\downarrow$$

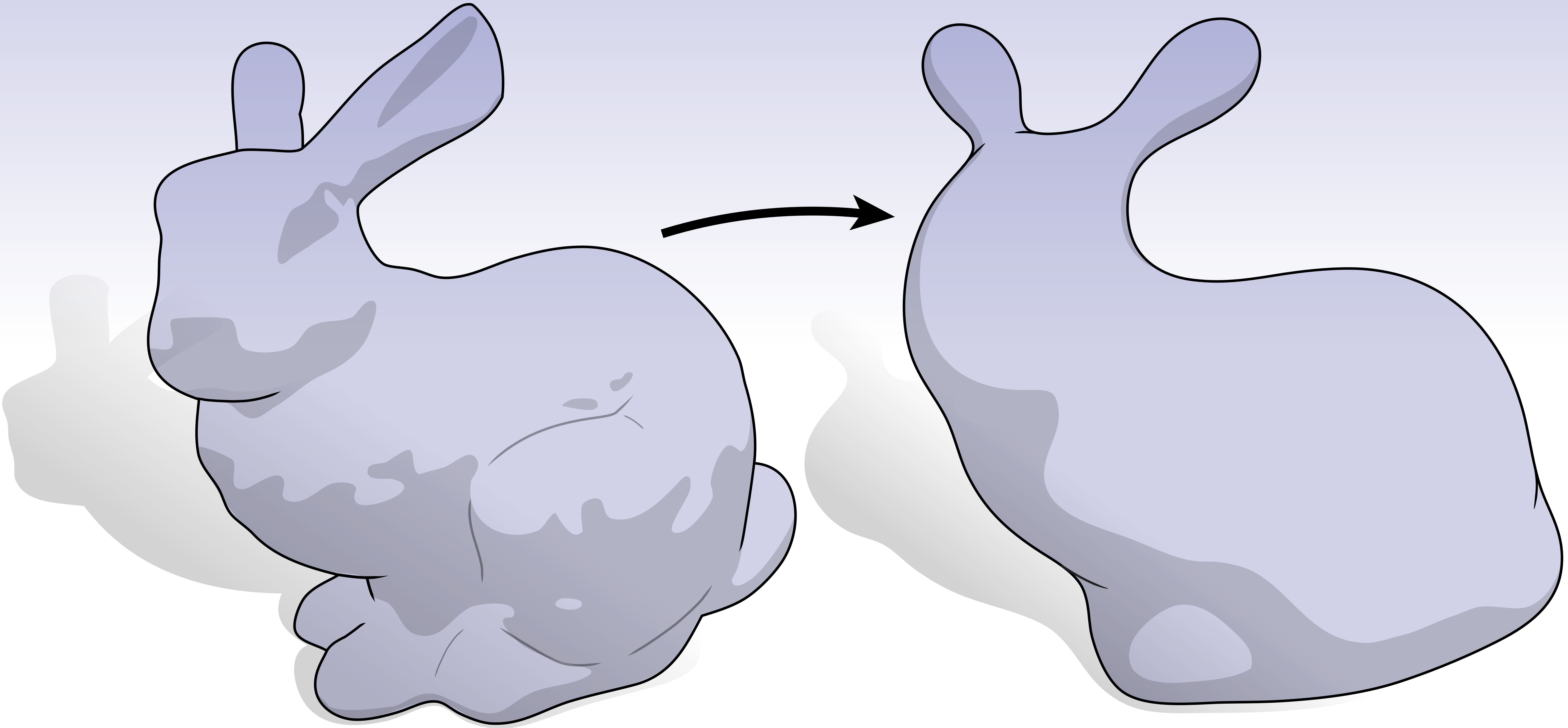
$$\left(\star_0 - h d_0^t \star_1 d_0\right) f = \star_0 f_0$$

Desbrun et al. 1999

Pseudo-code

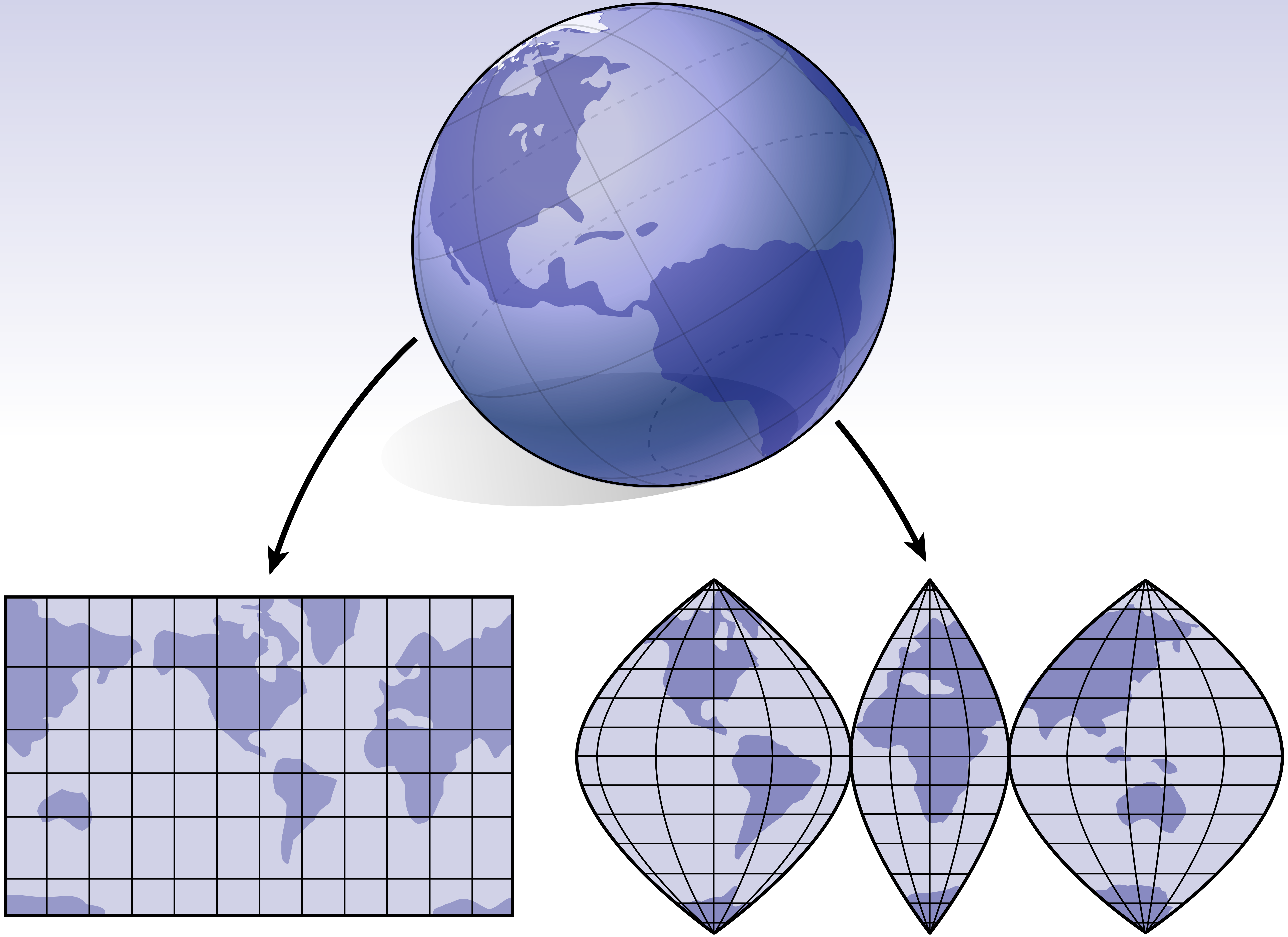
```
// current points
DenseMatrix<Real> x0;
getPositions(mesh, x0);

// build matrix
SparseMatrix<Real> L = d0^t * star1 * d0;
SparseMatrix<Real> A = star0 + h * L;

// linear system
DenseMatrix<Real> x0;
solve(A, x, star0 * x0);

// new points
setPositions(x, mesh);
```

Example

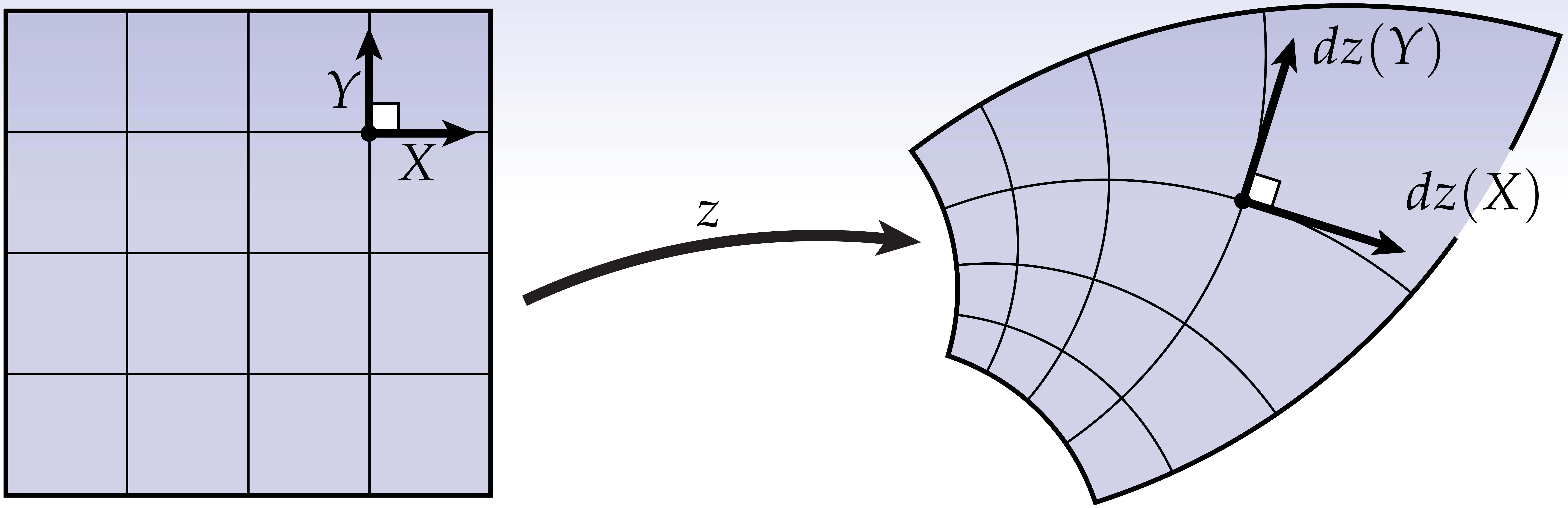


Parametrization

Problem Statement



Conformal Parametrization



Cauchy-Riemann equation

Cauchy-Riemann equation

$$dz(N \times X) = i\,dz(X)$$

Cauchy-Riemann equation

$$dz(N \times X) = i\,dz(X)$$
$$\star dz(X) = i\,dz(X)$$

Cauchy-Riemann equation

$$dz(N \times X) = i\,dz(X)$$
$$\star dz(X) = i\,dz(X)$$
$$\star dz = i\,dz$$

Conformal Energy

$$E_c(z) := \frac{1}{4}\|\star\, dz - i dz\|^2$$

Conformal Energy

$$E_c(z) := \frac{1}{4}\|\star\, dz - i dz\|^2 = \frac{1}{2}\langle \Delta z, z\rangle - \mathrm{Area}(z)$$

Conformal Energy

$$E_c(z) := \frac{1}{4}\|\star dz - i dz\|^2 = \frac{1}{2}\langle \Delta z, z\rangle - \mathrm{Area}(z) = \frac{1}{2}\langle L_c z, z\rangle$$

Conformal Energy

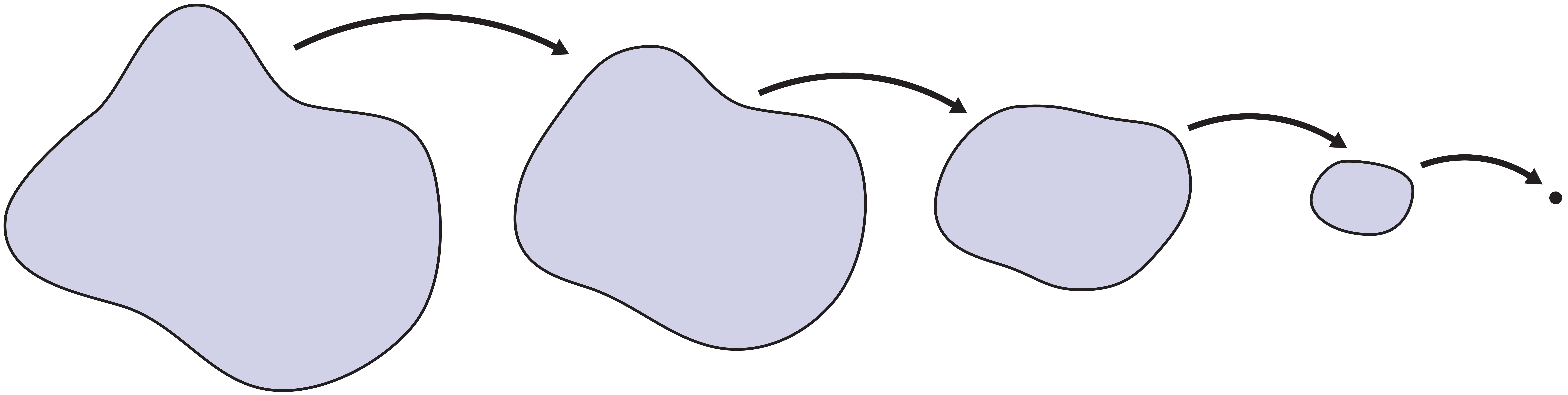
$E_c(z) := \frac{1}{4}\|\star dz - i dz\|^2 = \frac{1}{2}\langle \Delta z, z\rangle - \mathrm{Area}(z) = \frac{1}{2}\langle L_c z, z\rangle$

$\min_z E_c(z)$

Conformal Energy

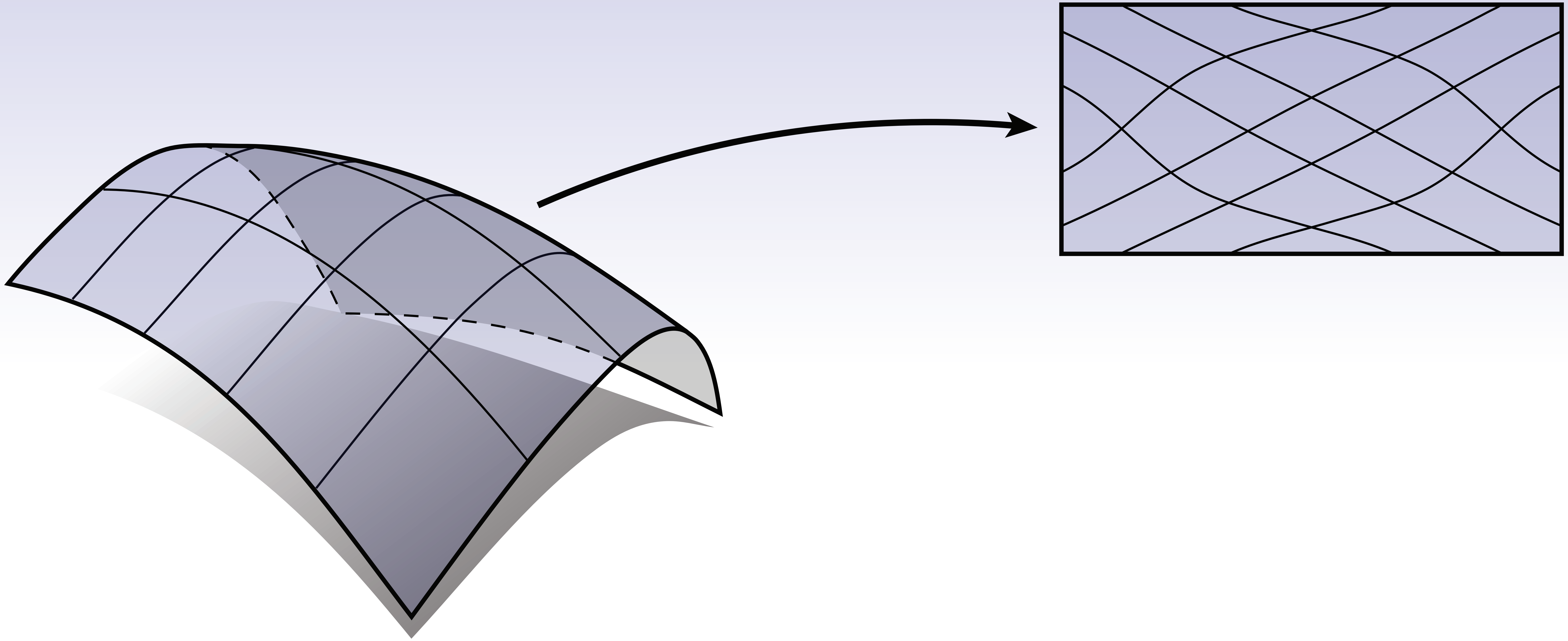
$$E_c(z) := \frac{1}{4}\|\star dz - i dz\|^2 = \frac{1}{2}\langle \Delta z, z\rangle - \mathrm{Area}(z) = \frac{1}{2}\langle L_c z, z\rangle$$

$$\min_z E_c(z)$$

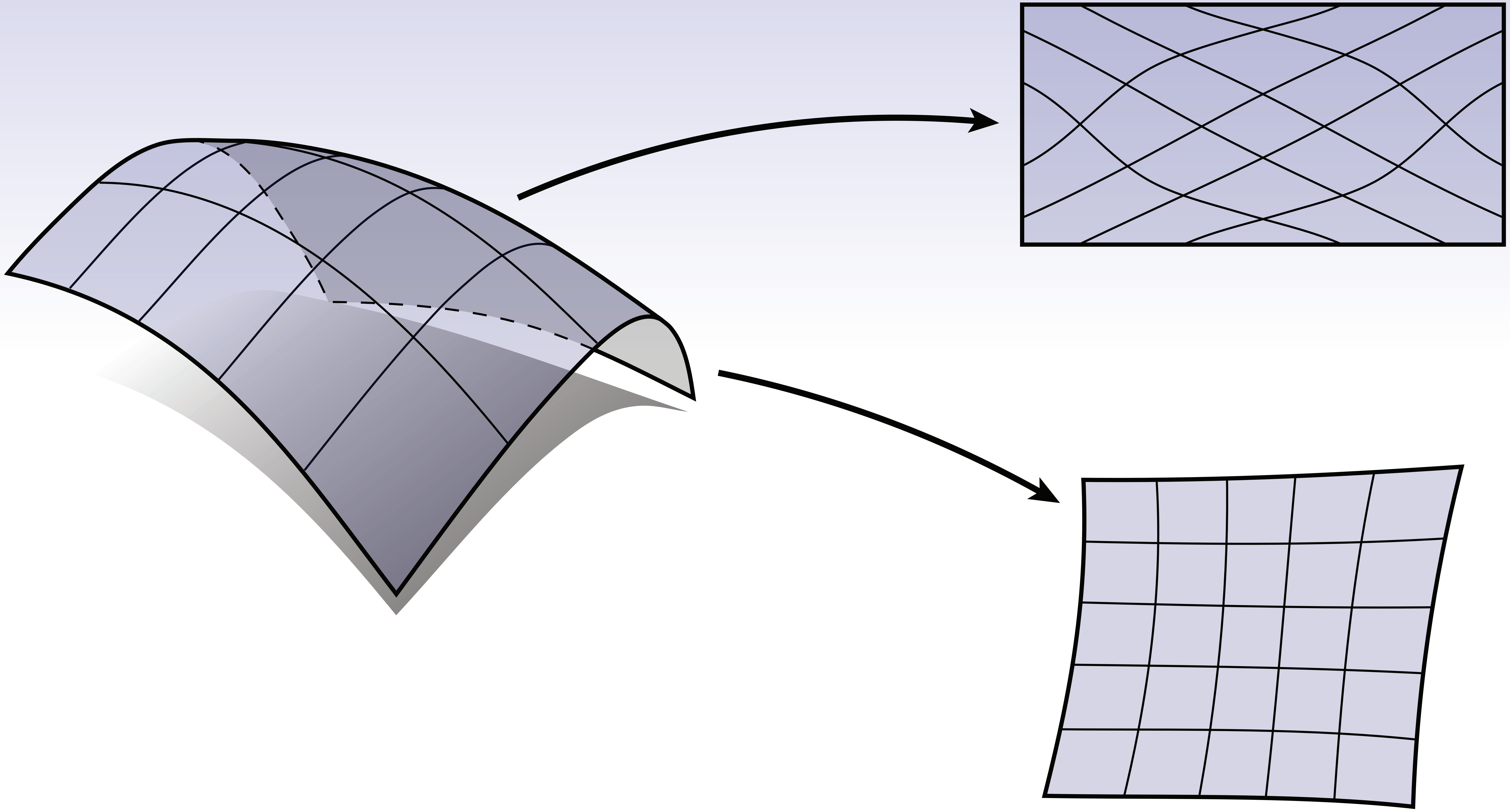


Constraints

Constraints



Constraints



Spectral approach

Mullen et al. 2008

Spectral approach

- remove translation $\langle z, \mathbf{1} \rangle = 0$

Mullen et al. 2008

Spectral approach

- remove translation $\langle z, \mathbf{1} \rangle = 0$
- remove scale $\langle z, z \rangle = 1$

Mullen et al. 2008

Spectral approach

- remove translation $\langle z, \mathbf{1} \rangle = 0$
- remove scale $\langle z, z \rangle = 1$

$$\min_z E_c(z)$$
$$\langle z, \mathbf{1} \rangle = 0$$
$$\langle z, z \rangle = 1$$

Mullen et al. 2008

Spectral approach

- remove translation $\langle z, \mathbf{1} \rangle = 0$
- remove scale $\langle z, z \rangle = 1$

$$\begin{array}{l} \min_z E_c(z) \\ \langle z, \mathbf{1} \rangle = 0 \\ \langle z, z \rangle = 1 \end{array} \longleftrightarrow \mathrm{L}_c z = \lambda_2 z$$

Mullen et al. 2008

Pseudo-code

```
// L
SparseMatrix<Complex> L = d0^t * star1 * d0;

// Lc = L - A
foreach face f:
   foreach edge (vi,vj):
   Lc(vi,vj) = L(vi,vj) - 0.5 * i;
   Lc(vj,vi) = L(vi,vj) + 0.5 * i;

// eigenvalue problem
DenseMatrix<Complex> x;
smallestEigenvalue(Lc, x);
```

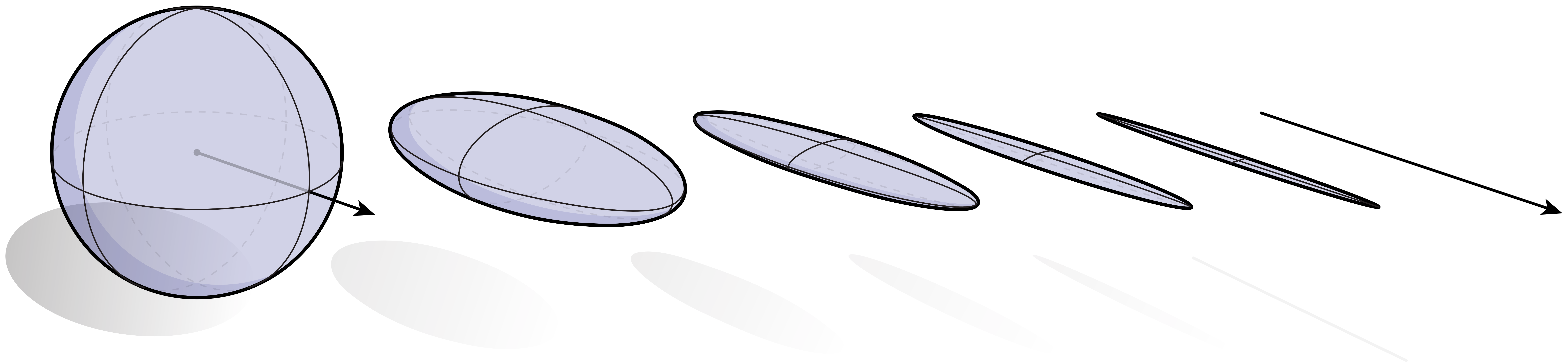
Power method

Power method

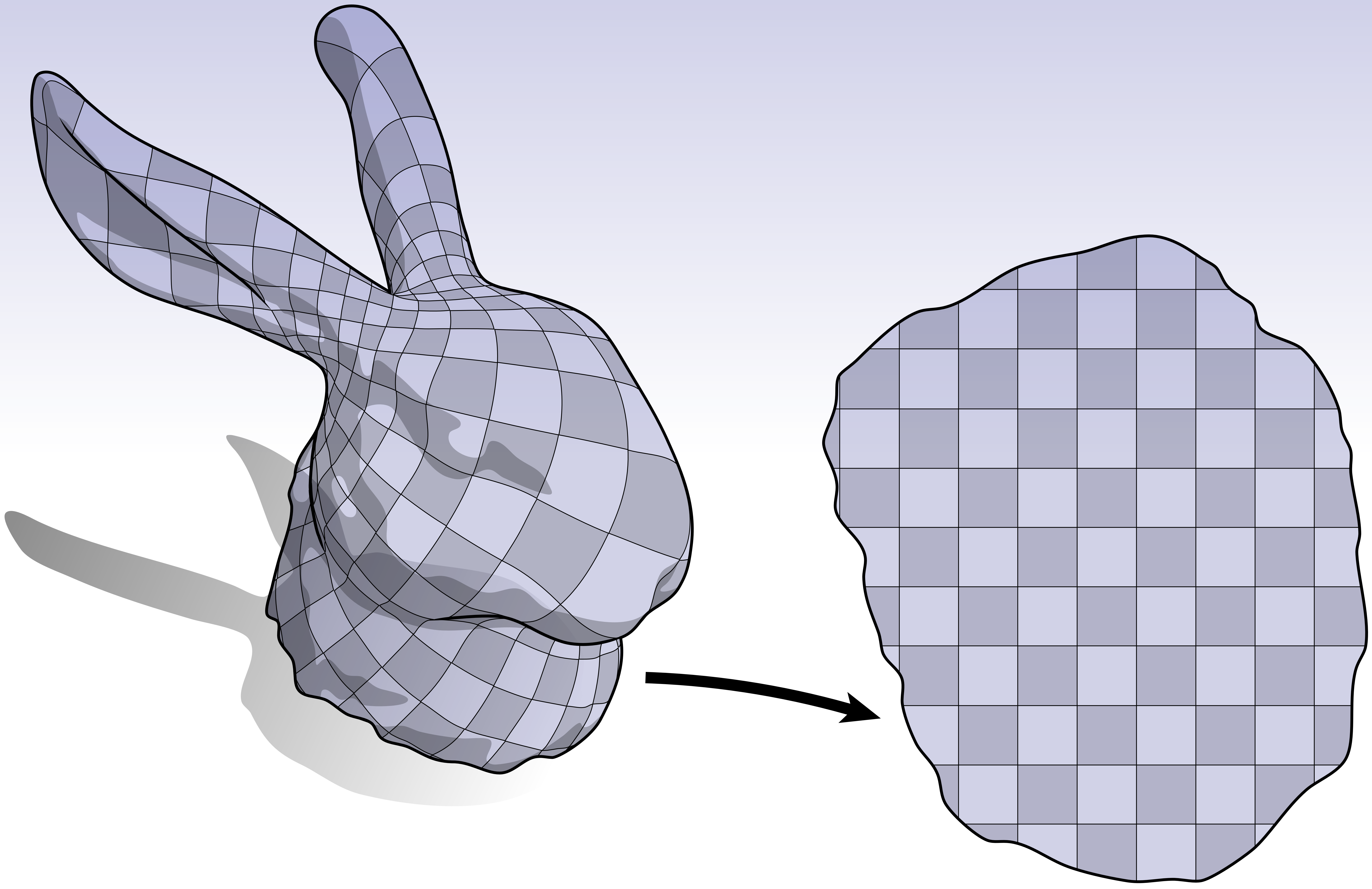
```
function smallestEigenvalue( Lc, x ):
   factorize(Lc);
   x.randomize();
   until convergence:
      solve( Lc, x, x );
      x = remove_mean( x );
      x = normalize( x );
```

Power method

```
function smallestEigenvalue( Lc, x ):
   factorize(Lc);
   x.randomize();
   until convergence:
      solve( Lc, x, x );
      x = remove_mean( x );
      x = normalize( x );
```



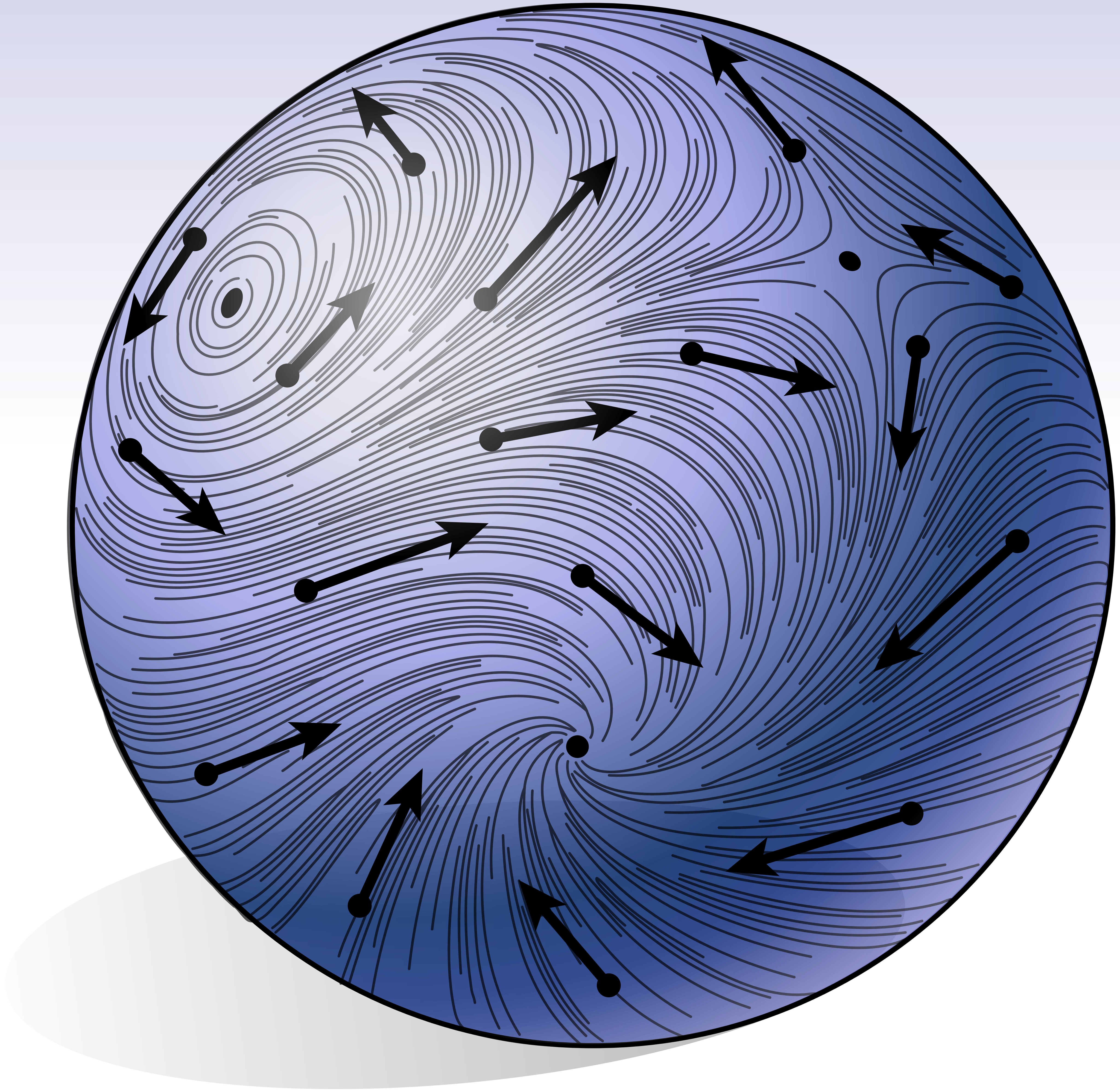
Example



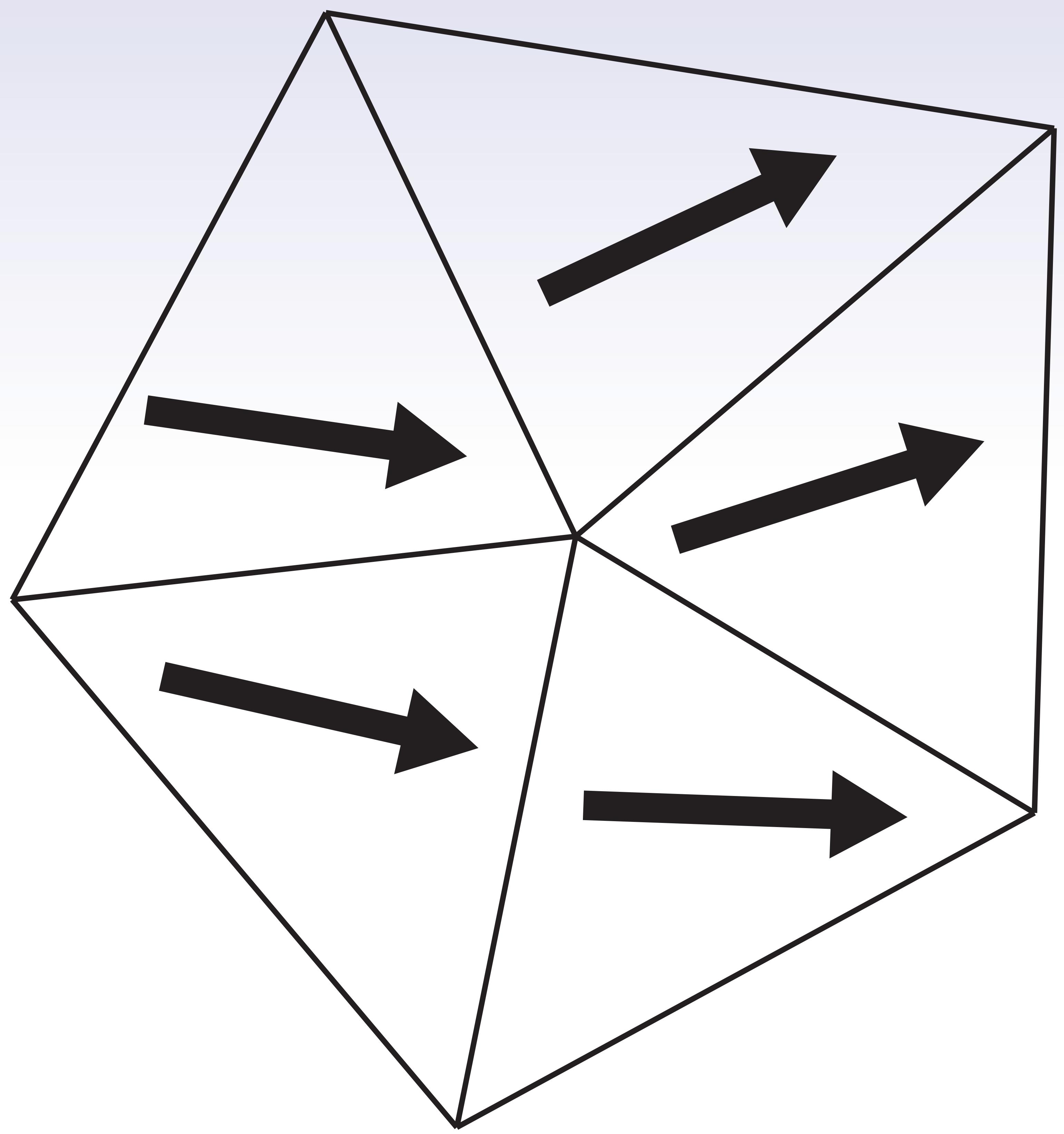
Vector field design

Problem Statement

course notes: chapter 8
code / Connection

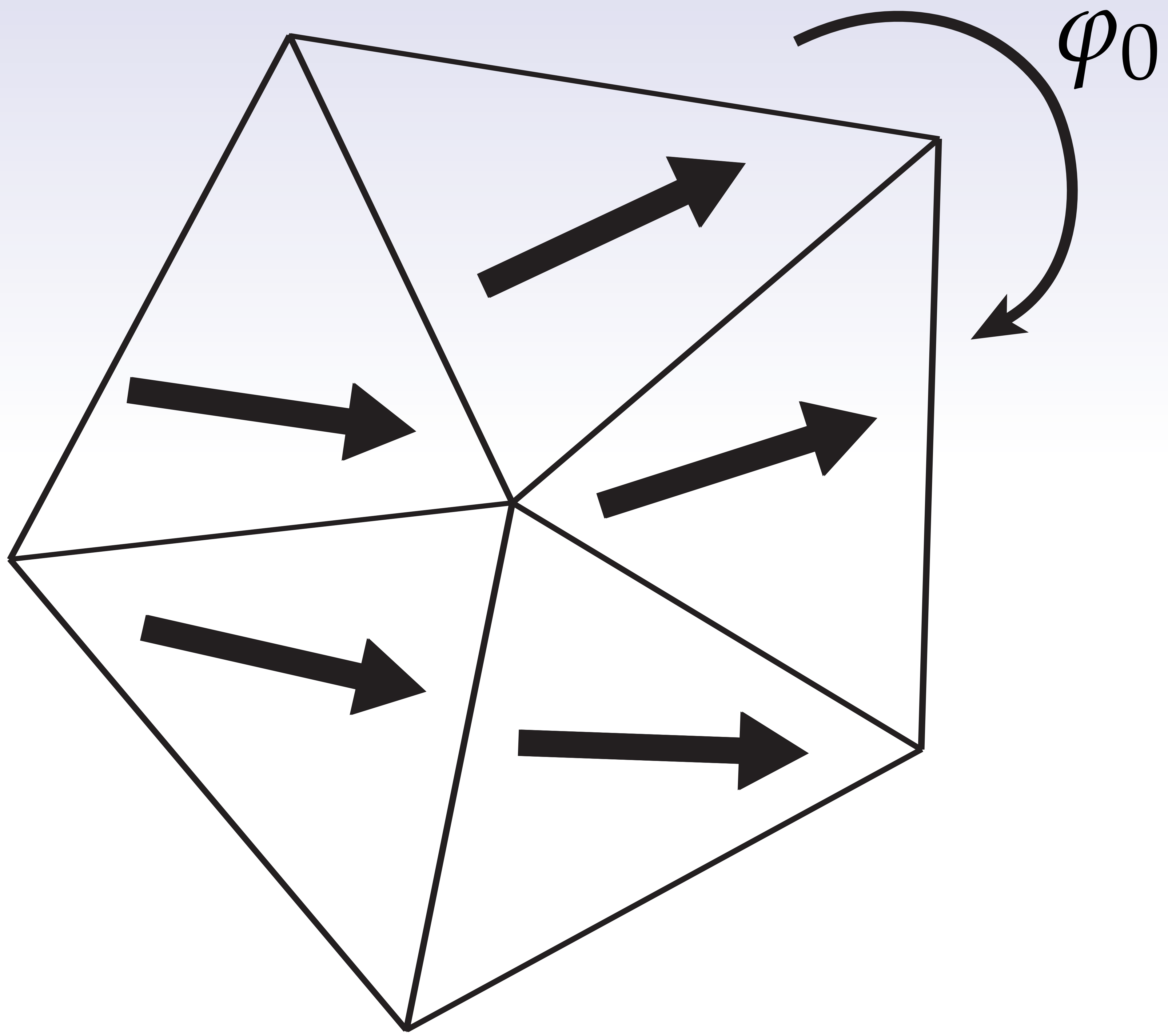


Discrete Connection



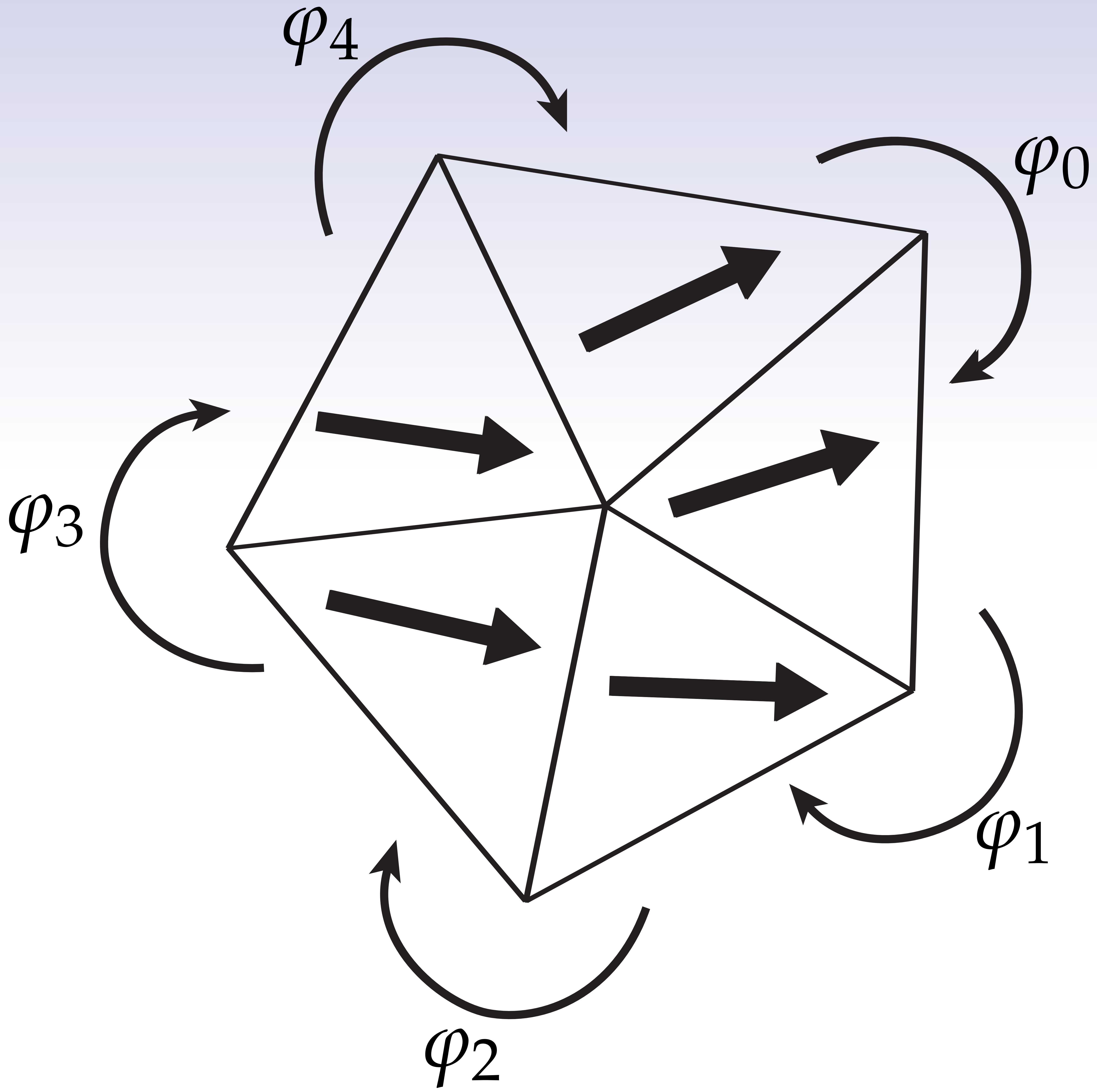
Crane et al. 2010

Discrete Connection



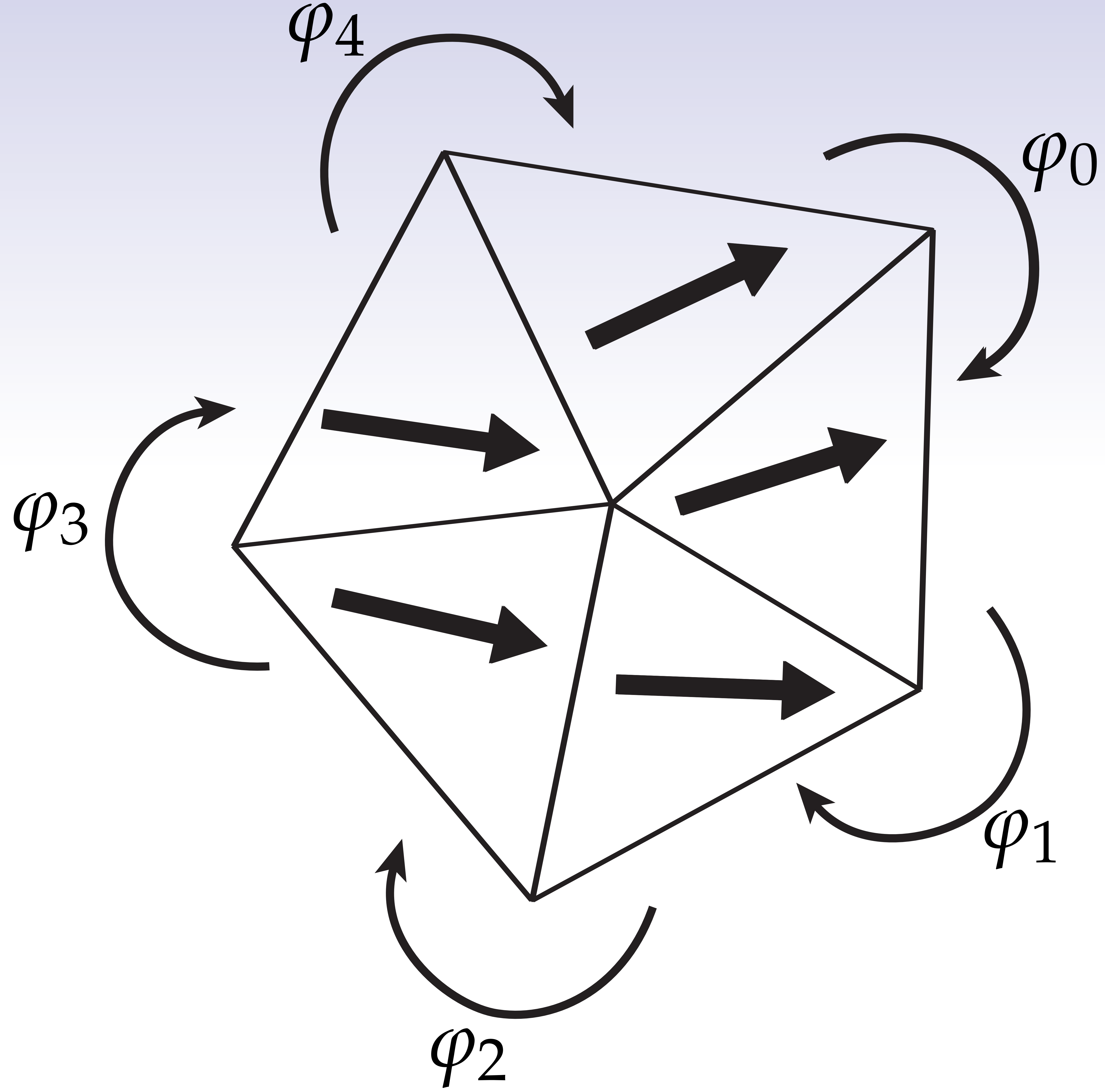
Crane et al. 2010

Discrete Connection



Crane et al. 2010

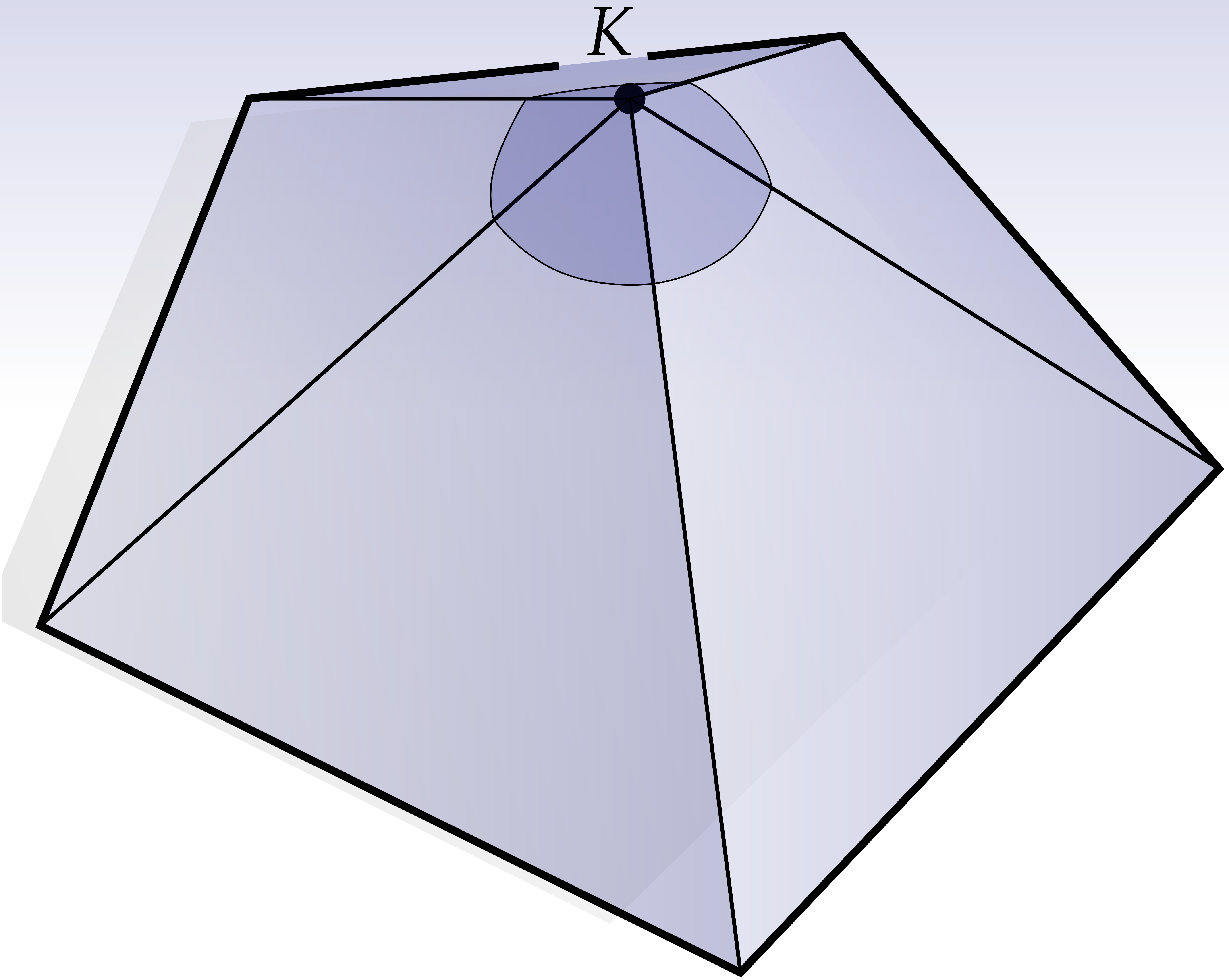
Discrete Connection



$$d\varphi = 2\pi k$$

Crane et al. 2010

Gaussian Curvature



$$d\varphi = -K + 2\pi k$$

Least Twisting Connection

$$\min_{\varphi} \|\varphi\|^2$$

$$d\varphi = -K + 2\pi k$$

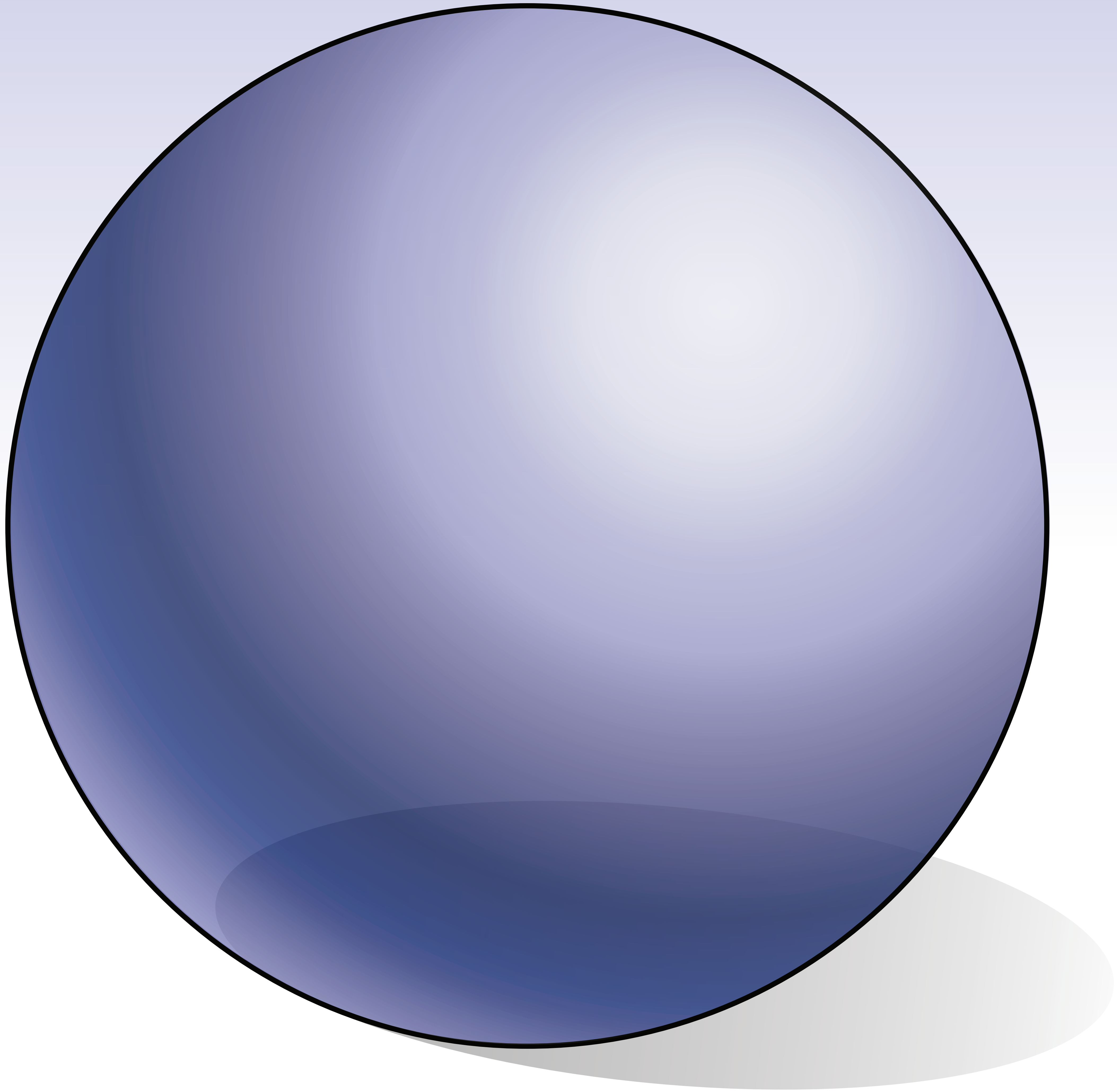
Least Twisting Connection

$$\min_\varphi \|\varphi\|^2$$
$$d\varphi = -K + 2\pi k$$

de Goes and Crane 2010

Least Twisting Connection

$$\min_{\varphi} \|\varphi\|^2$$
$$d\varphi = -K + 2\pi k$$



de Goes and Crane 2010

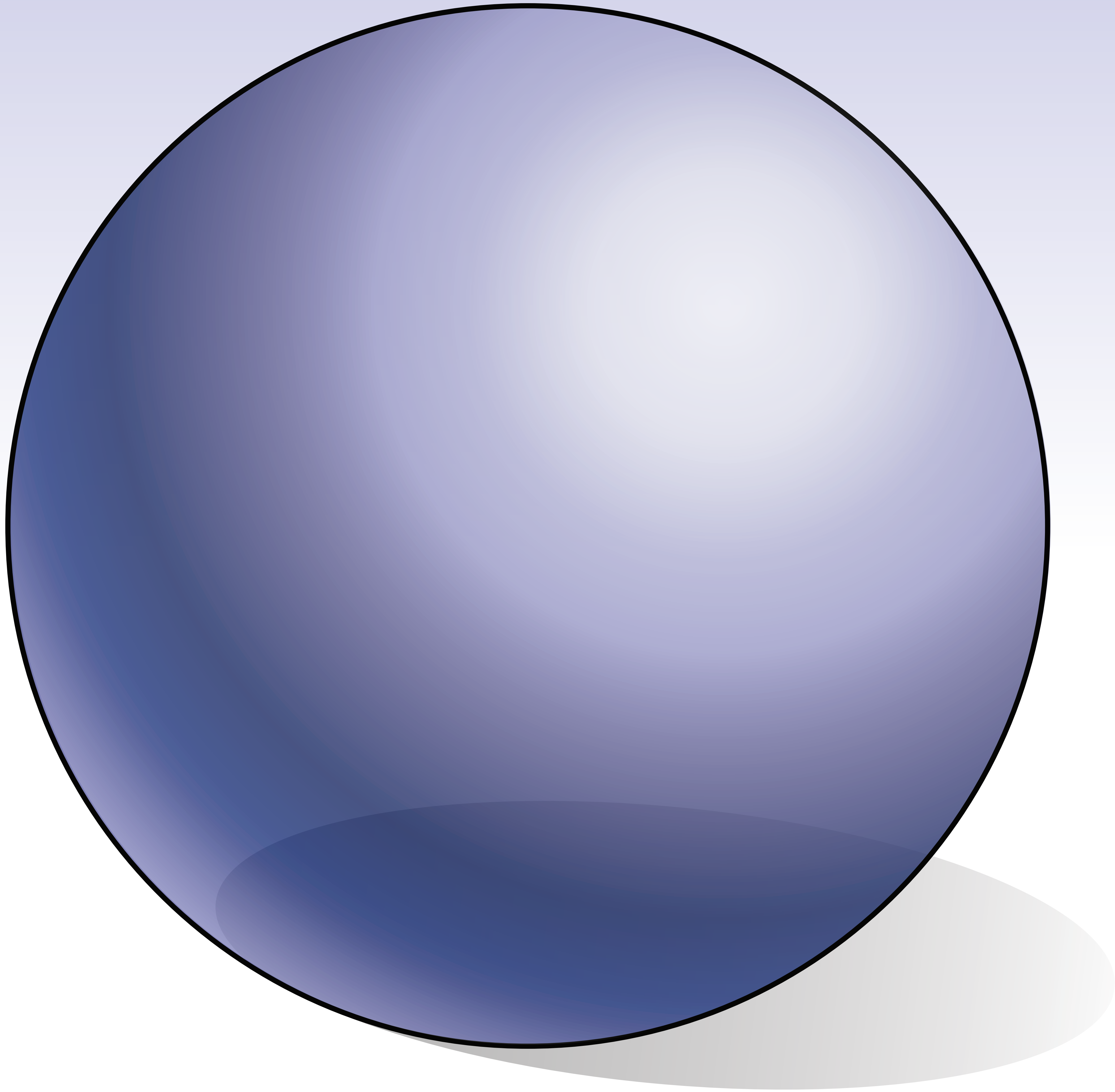
Least Twisting Connection

$\min_{\varphi} \|\varphi\|^2$

$d\varphi = -K + 2\pi k$

$$\Big\downarrow \; \varphi = d\alpha + \delta\beta$$

$\Delta\beta = -K + 2\pi k$

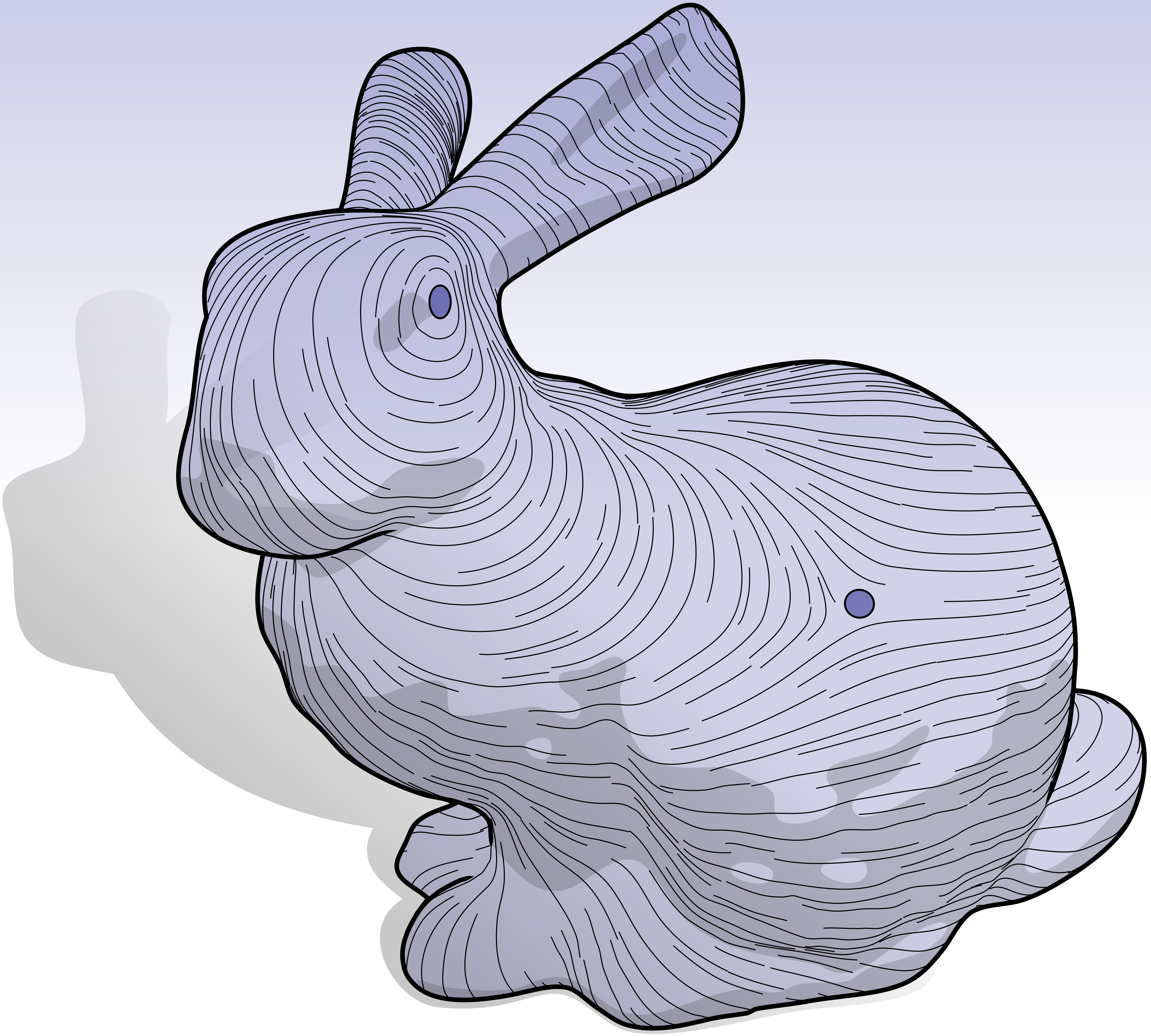


de Goes and Crane 2010

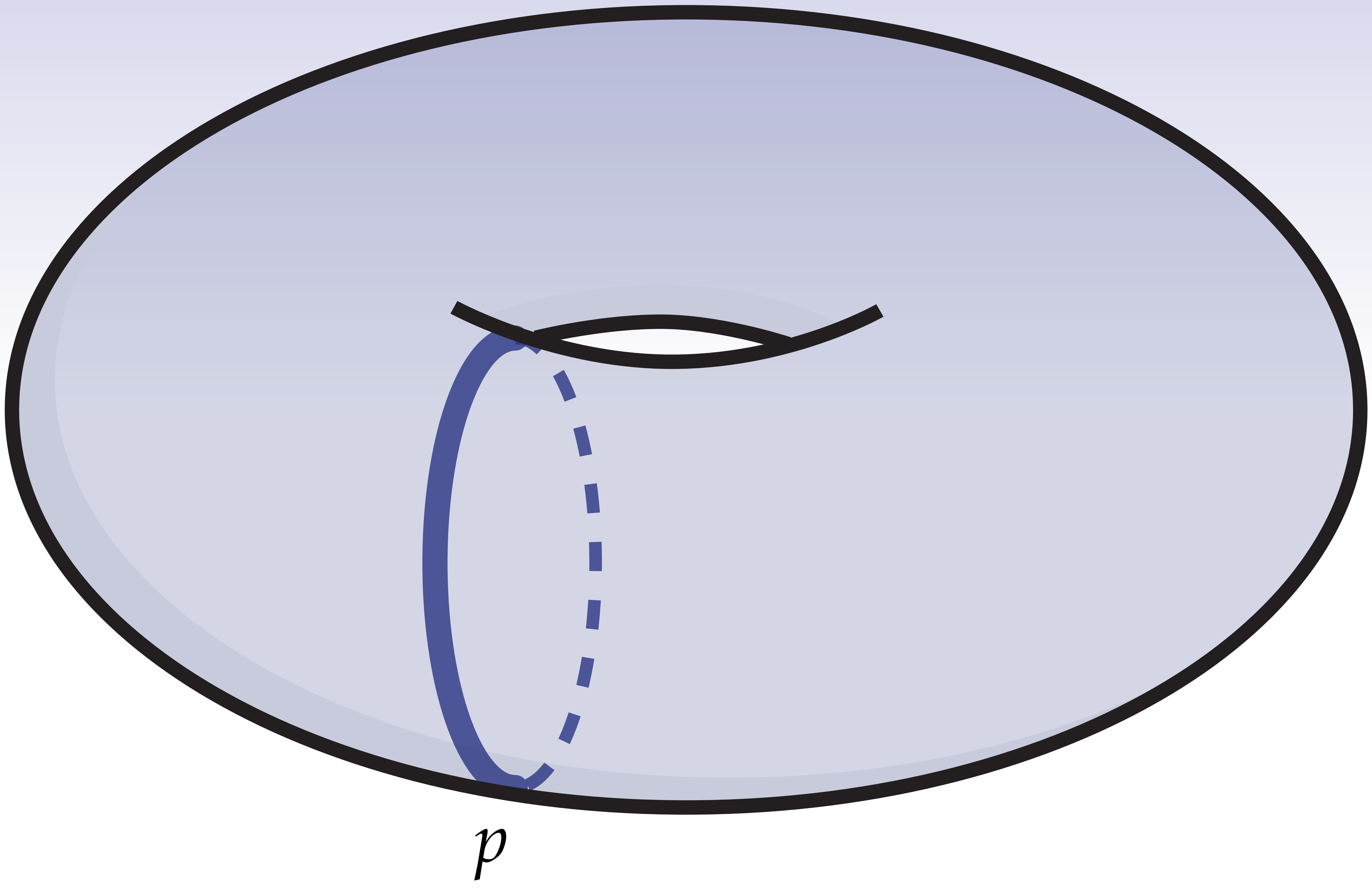
Pseudo-code

```
// build matrix
SparseMatrix<Real> L = d0^t * star1 * d0;

// build rhs
DenseMatrix<Real> b(#Vertices);
foreach vertex v:
   b(v) = 2*PI*k - curvature(v);

// linear system
DenseMatrix<Real> beta;
solve( L, beta, b );
```

Example



Arbitrary Topology



$$C\varphi = -p + 2\pi k$$

Least Twisting Connection++

$$\min_{\varphi} \|\varphi\|^2$$

$$d\varphi = -K + 2\pi k$$
$$C\varphi = -p + 2\pi k$$

Least Twisting Connection++

$$\min_\varphi \|\varphi\|^2$$

$$d\varphi = -K + 2\pi k$$
$$C\varphi = -p + 2\pi k$$

$$\Big\downarrow \; \varphi = d\alpha + \delta\beta + \textstyle\sum_i z_i \gamma_i$$

$$\Delta\beta = -K + 2\pi k$$
$$\mathrm{P}\mathrm{z} = -p + 2\pi k + C\delta\beta$$

Period matrix

$$\mathrm{P}_{ij} = \mathrm{C}_i \gamma_j$$

Least Twisting Connection++

$$\min_\varphi \|\varphi\|^2$$

$$d\varphi = -K + 2\pi k$$
$$C\varphi = -p + 2\pi k$$

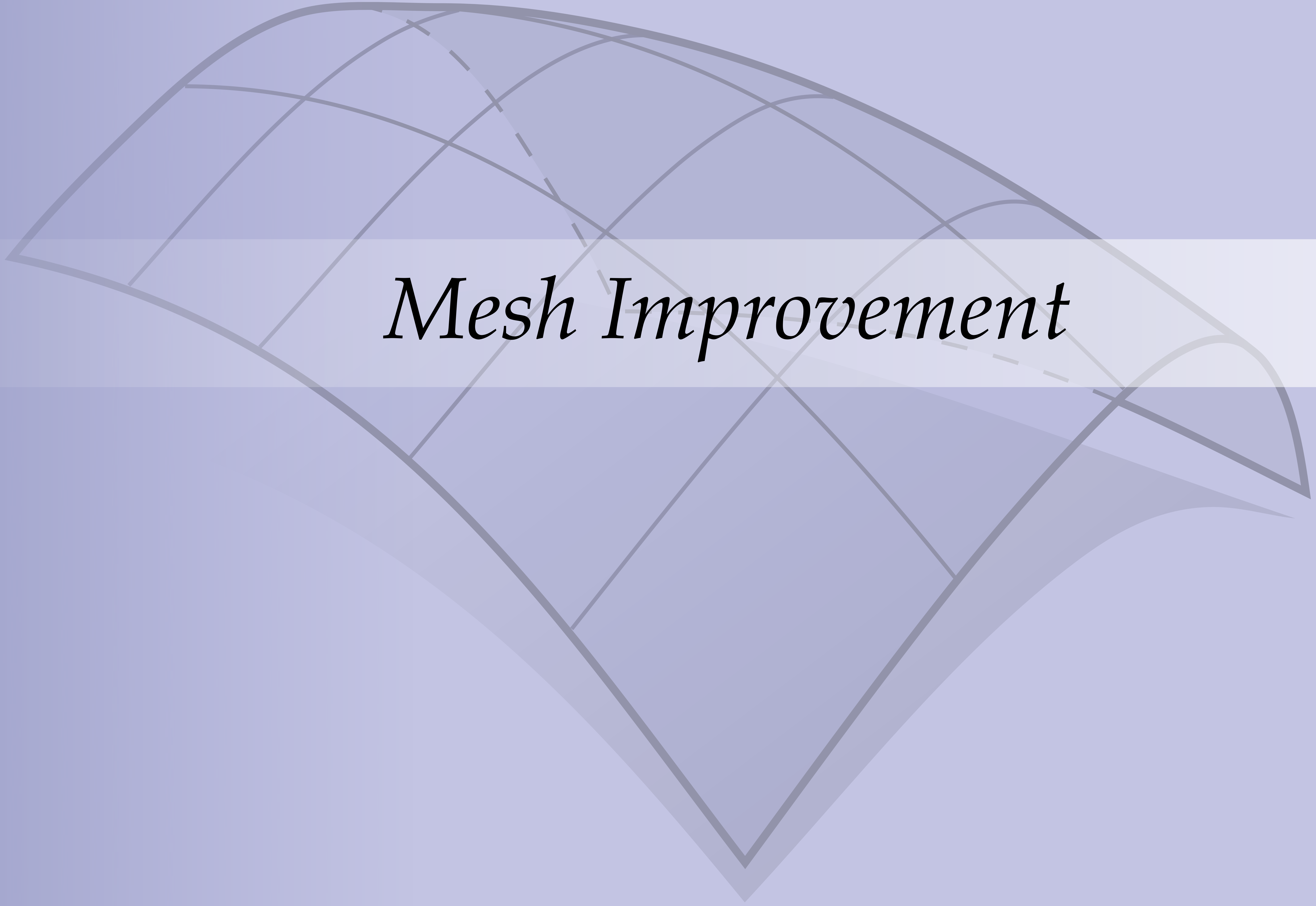
$$\varphi = d\alpha + \delta\beta + \textstyle\sum_i z_i \gamma_i$$

$$\Delta\beta = -K + 2\pi k$$
$$\mathrm{P}z = -p + 2\pi k + C\delta\beta$$

Period matrix

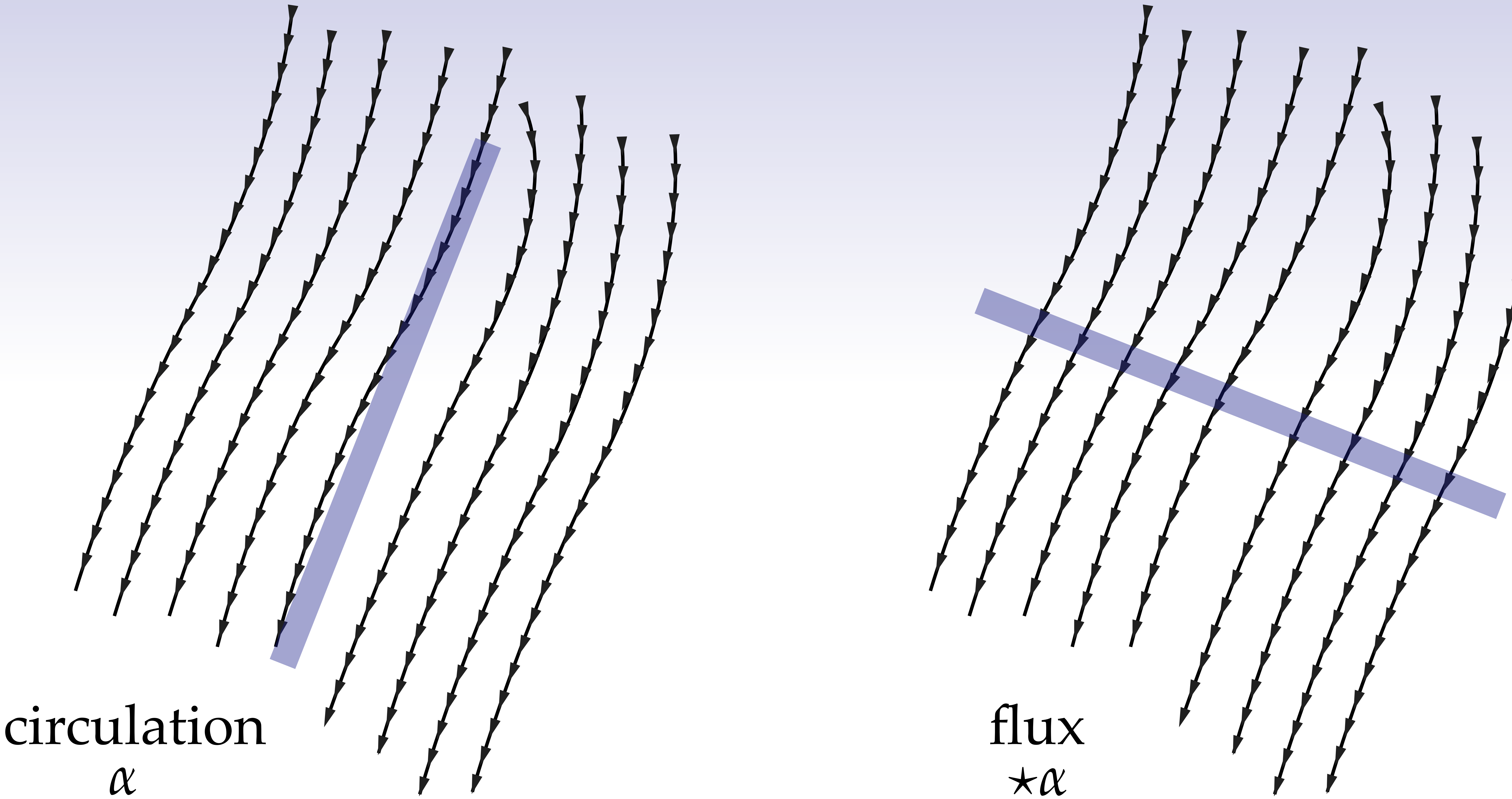
$$\mathrm{P}_{ij} = \mathrm{C}_i \gamma_j$$

```
// solve for beta
SparseMatrix<Real> L = d0^t * star1 * d0;

DenseMatrix<Real> K(#Vertices);
foreach vertex v:
   K(v) = curvature(v);

DenseMatrix<Real> beta;
solve( L, beta, b );

// solve for z
DenseMatrix<Real>  p(#Generators);
SparseMatrix<Real> P(#Generators, #Generators);
foreach generator g:
   foreach harmonic gamma:
      foreach edge e in g:
         P(g, gamma) += gamma(e);
   p(g) = generator_holonomy(g);

DenseMatrix<Real> z;
solve( P, z, p );
```

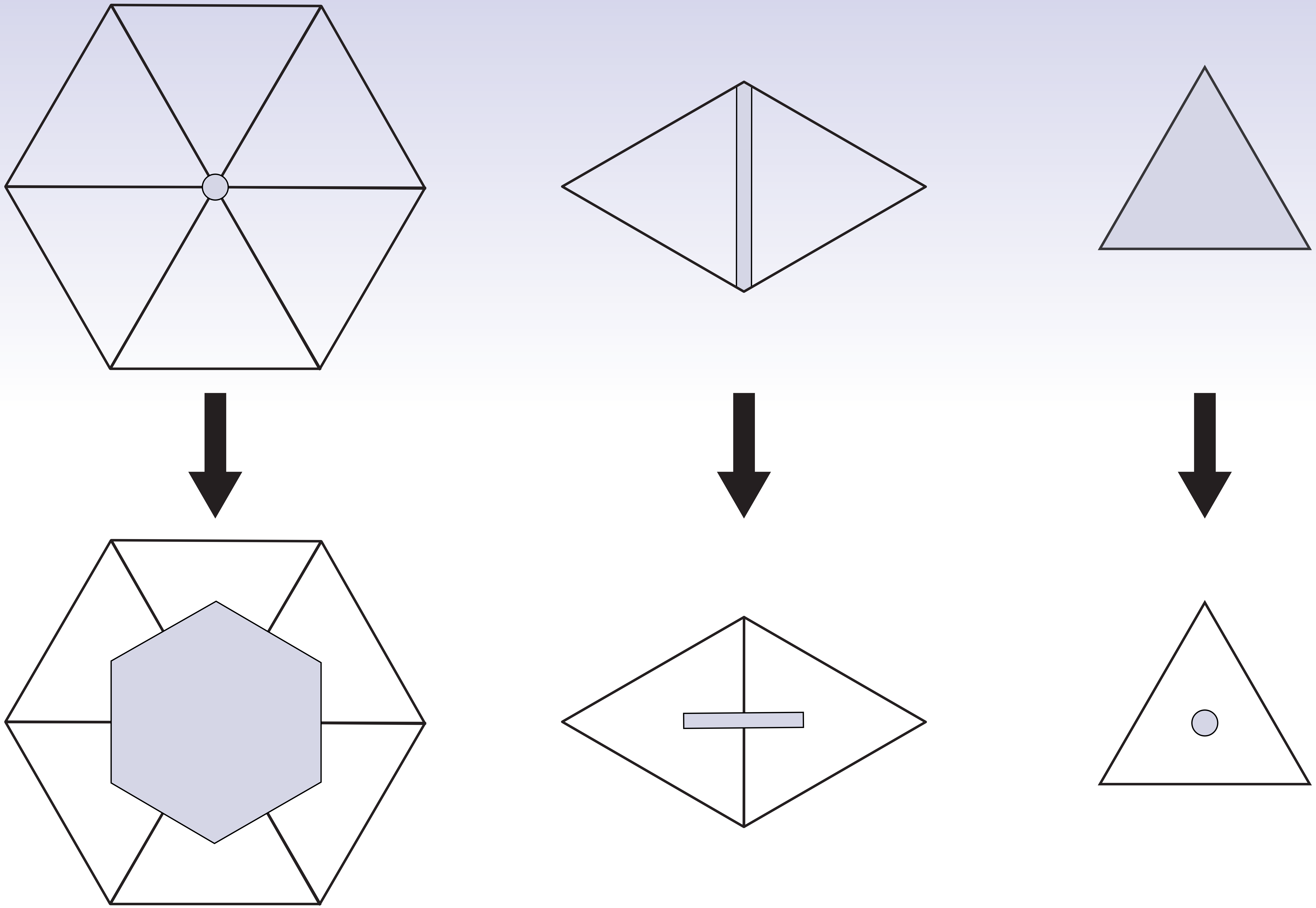


Mesh Improvement

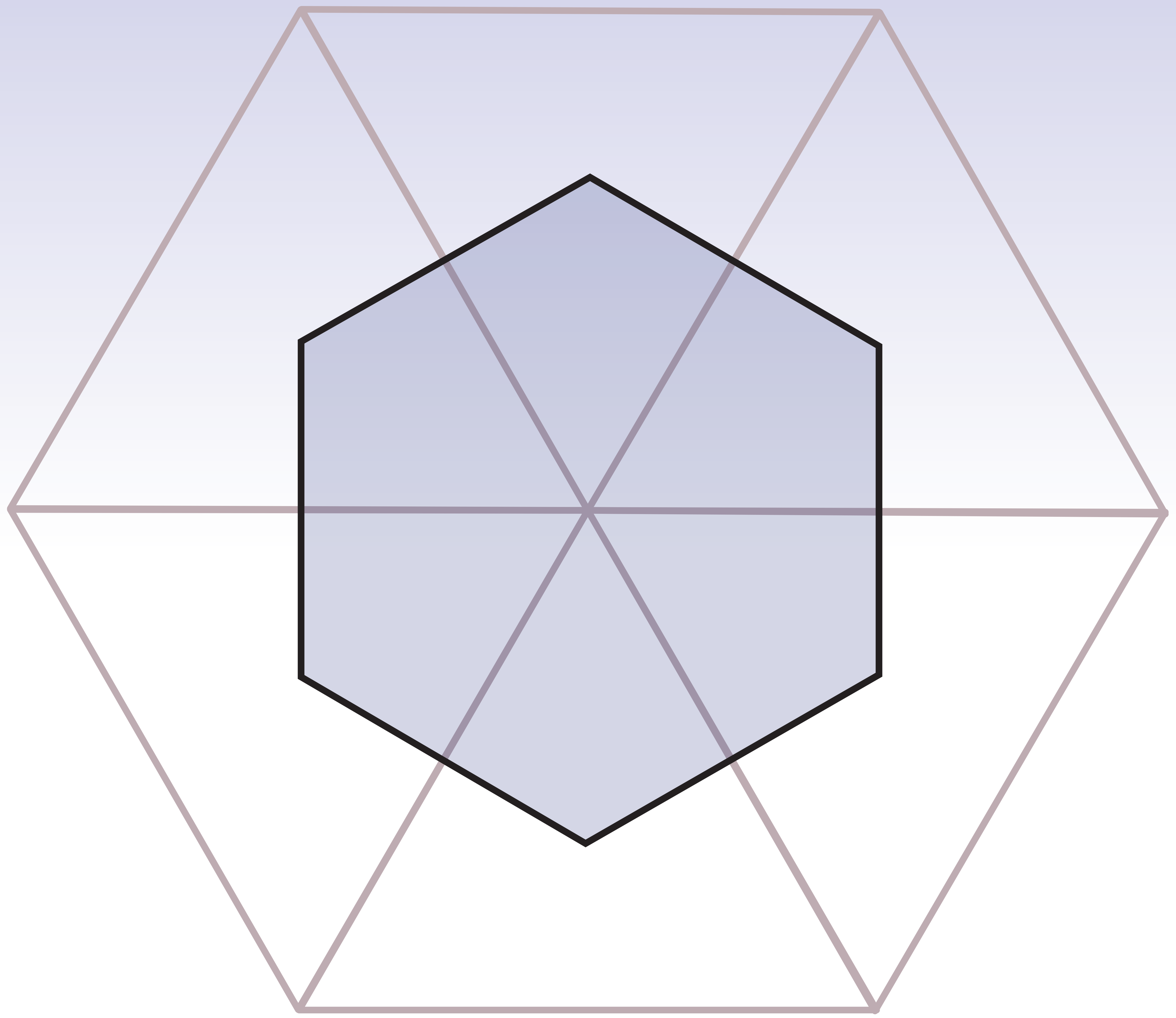
Duality



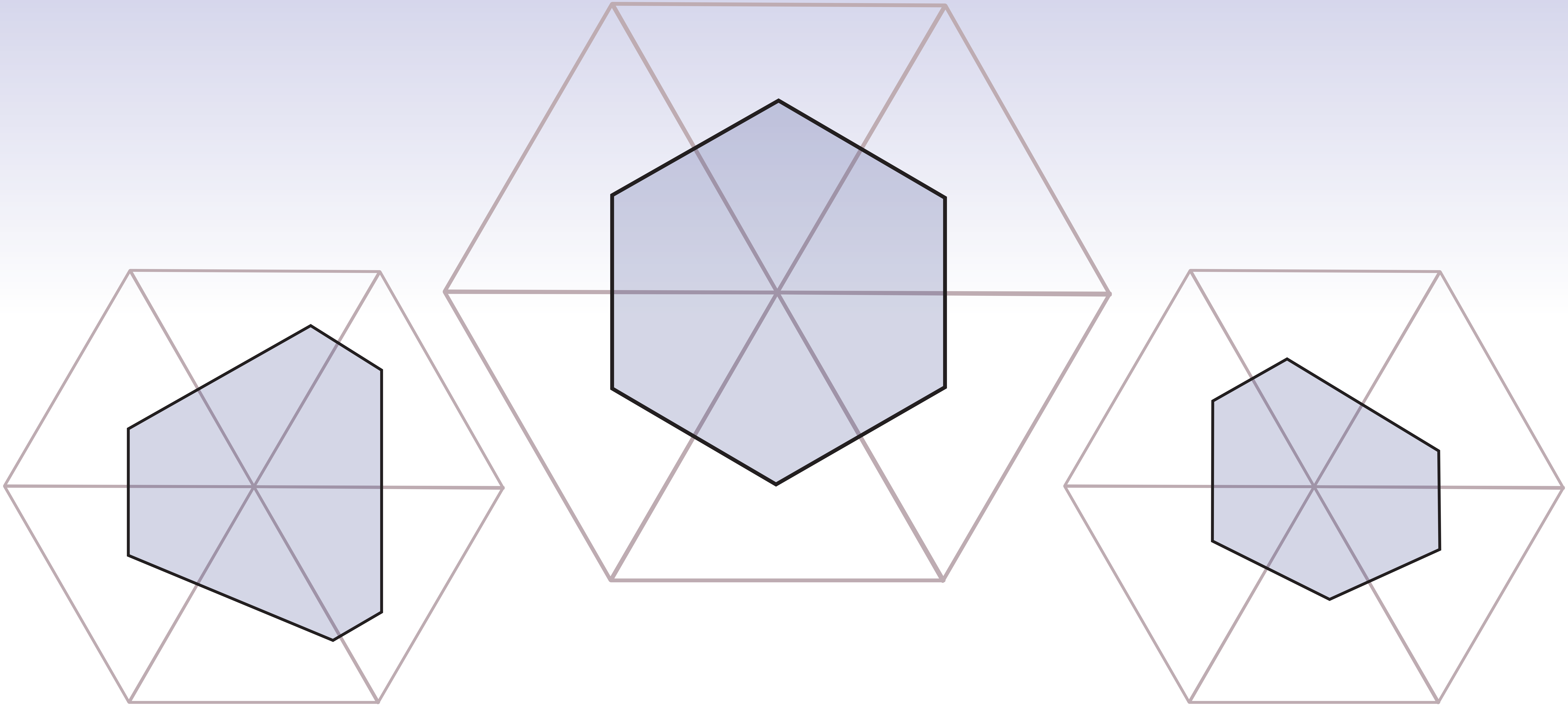
Orthogonal primal-dual meshes



Problem Statement



Problem Statement



In Siggraph 2013

On the Equilibrium of Simplicial Masonry Structures

Fernando de Goes, Caltech
Pierre Alliez, INRIA
Houman Owhadi, Caltech
Mathieu Desbrun, Caltech

Abstract

We present a novel approach for the analysis and design of self-supporting simplicial masonry structures. A finite-dimensional formulation of their compressive stress field is derived, offering a new interpretation of thrust networks through numerical homogenization theory. We further leverage geometric properties of the resulting force diagram to identify a set of reduced coordinates characterizing the equilibrium of simplicial masonry. We finally derive computational form-finding tools that improve over previous work in efficiency, accuracy, and scalability.

Keywords: architectural geometry, discrete differential geometry, geometry processing, orthogonal reciprocal diagram.

Links: DL PDF WEB

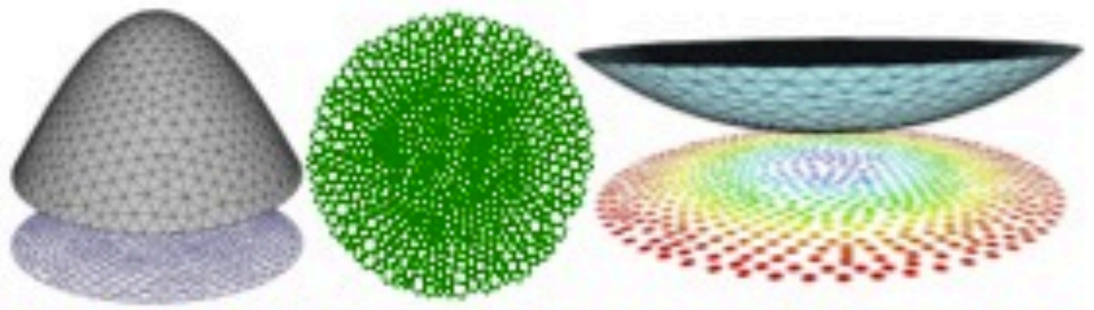


Figure 1: ***Simplicial Masonry.*** *A self-supporting simplicial masonry structure is a triangle mesh, defined as a height field over the plane (left). We show through numerical homogenization of the stress tensor that they support their own weights if there exists an orthogonal dual diagram (middle), representing a finite-dimensional approximation of the purely compressive forces at play. We provide discrete counterparts to a number of continuous fields (such as the convex Airy stress function, right) and differential equations traditionally used in describing the equilibrium of these masonry buildings; in particular, we offer a set of reduced coordinates of the space of statically admissible shapes, linking our approach to regular triangulations.*

1 Introduction

> *The most subtle and exquisite part of architecture [...] is the formation of [...] vaults; cutting their stones, and adjusting them with such artifice, that the same gravity and weight that should have precipitated them to the ground, maintain them constantly in the air, supporting one another in virtue of the mutual complication which binds them [...].*
>
> Vicente Tosca, *Compendio Matematico (vol. 5-15)*, 1727

Masonry structures are arrangements of material blocks, such as bricks or stones, that support their own weight. Constructing curved vaults or domes with compression-only structures of blocks, further prevented from slipping through friction and/or mortar, has been practiced since antiquity. It is therefore no surprise that form finding and stability analysis of masonry structures have been an active area of research for years.

Equilibrium of a masonry structure is ensured if there exists an inner *thrust* surface which forms a compressive membrane resisting the external loads [Heyman 1966]. Balance conditions relating the stress field on the thrust surface to the loads are well understood in the continuous setting [Giaquinta and Giusti 1985; Fosdick and Schuler 2003; Angelillo et al. 2012]. Discretizing these equations have been

of form finding for masonry structures: while previous finite element methods are known to restrict the topology of masonry structures to the case of simply connected domains, thrust network approaches may lead to overconstrained balance equations depending on the choice of boundary conditions. To overcome these issues, we introduce in this paper a discrete theory of *simplicial masonry structures*. We show that the self-supporting properties of discrete simplicial structures can be derived from a numerical homogenization of the underlying continuous differential equations. By leveraging previous methods, we offer a unified computational framework that enforces the compressive nature and the equilibrium of masonry structures exactly, for surfaces of arbitrary topology. In the process, we introduce reduced coordinates to generate all possible reciprocal force diagrams from simplicial meshes, and reveal geometric connections to well-known continuous notions such as the Airy stress function. Finally, we turn our theoretical contributions into an effective computational technique for the design of simplicial masonry structures that offers improved performance over previous work.

Computing Self-Supporting Surfaces by Regular Triangulation

Yang Liu[*] Hao Pan[†] John Snyder[‡] Wenping Wang[†] Baining Guo[*]
[*]Microsoft Research Asia [†]The University of Hong Kong [‡]Microsoft Research

Figure 1: *Left: self-supporting surfaces with unsupported (top) and supported (bottom) boundary constraints. Unsupported boundary vertices and their corresponding power cells are colored in orange. Top right: initial self-supporting mesh. Spikes appear due to extremely small reciprocal areas. Bottom right: applying our smoothing scheme (5 iterations) improves mesh quality. The power diagrams (black) show how power cell area is distributed more evenly.*

Abstract

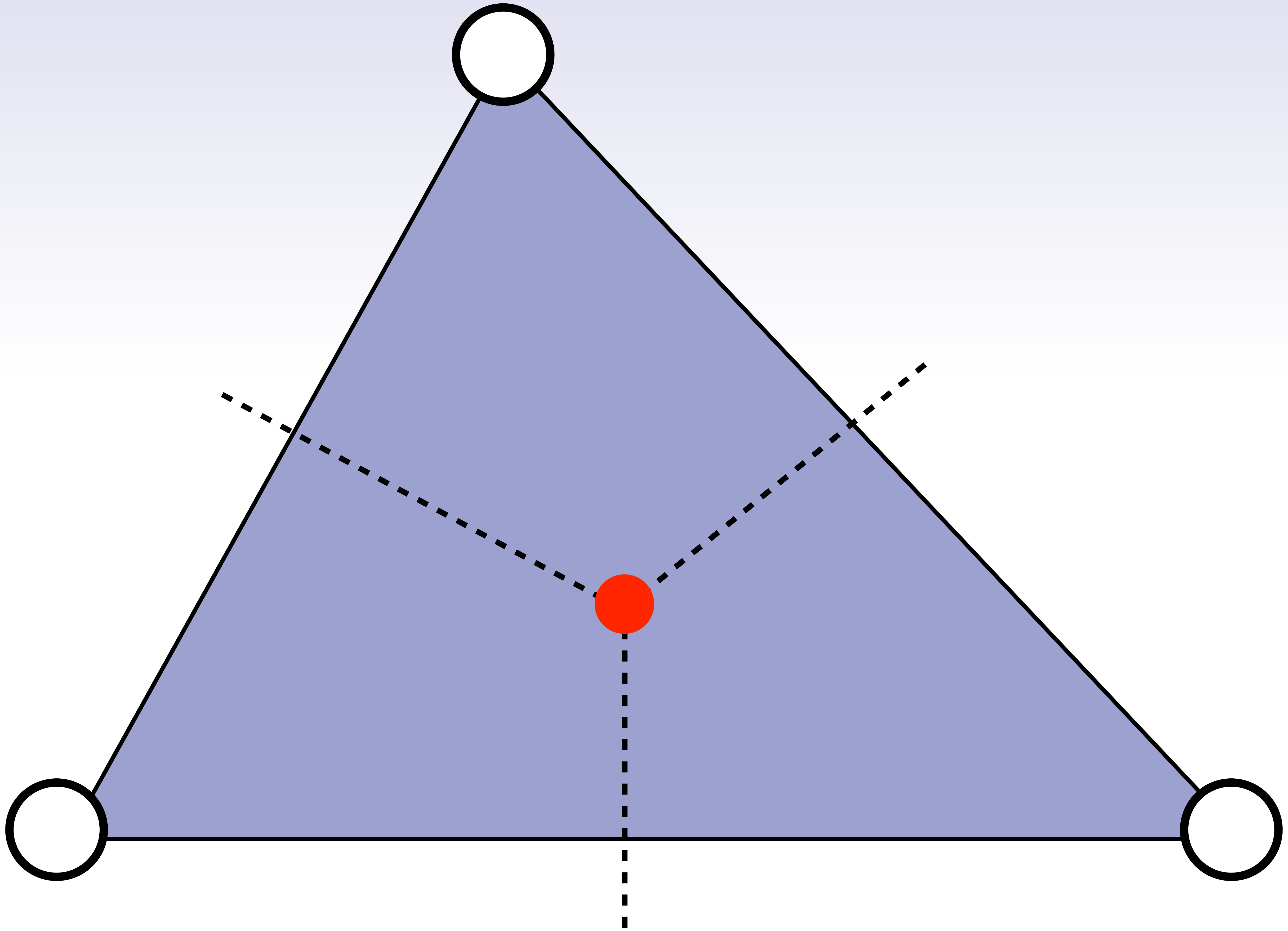
Masonry structures must be compressively self-supporting; designing such surfaces forms an important topic in architecture as well as a challenging problem in geometric modeling. Under certain conditions, a surjective mapping exists between a *power diagram*, defined by a set of 2D vertices and associated weights, and the reciprocal diagram that characterizes the force diagram of a discrete self-supporting network. This observation lets us define a new and convenient parameterization for the space of self-supporting networks. Based on it and the discrete geometry of this design space, we present novel geometry processing methods including surface smoothing and remeshing which significantly reduce the magnitude

1 Introduction

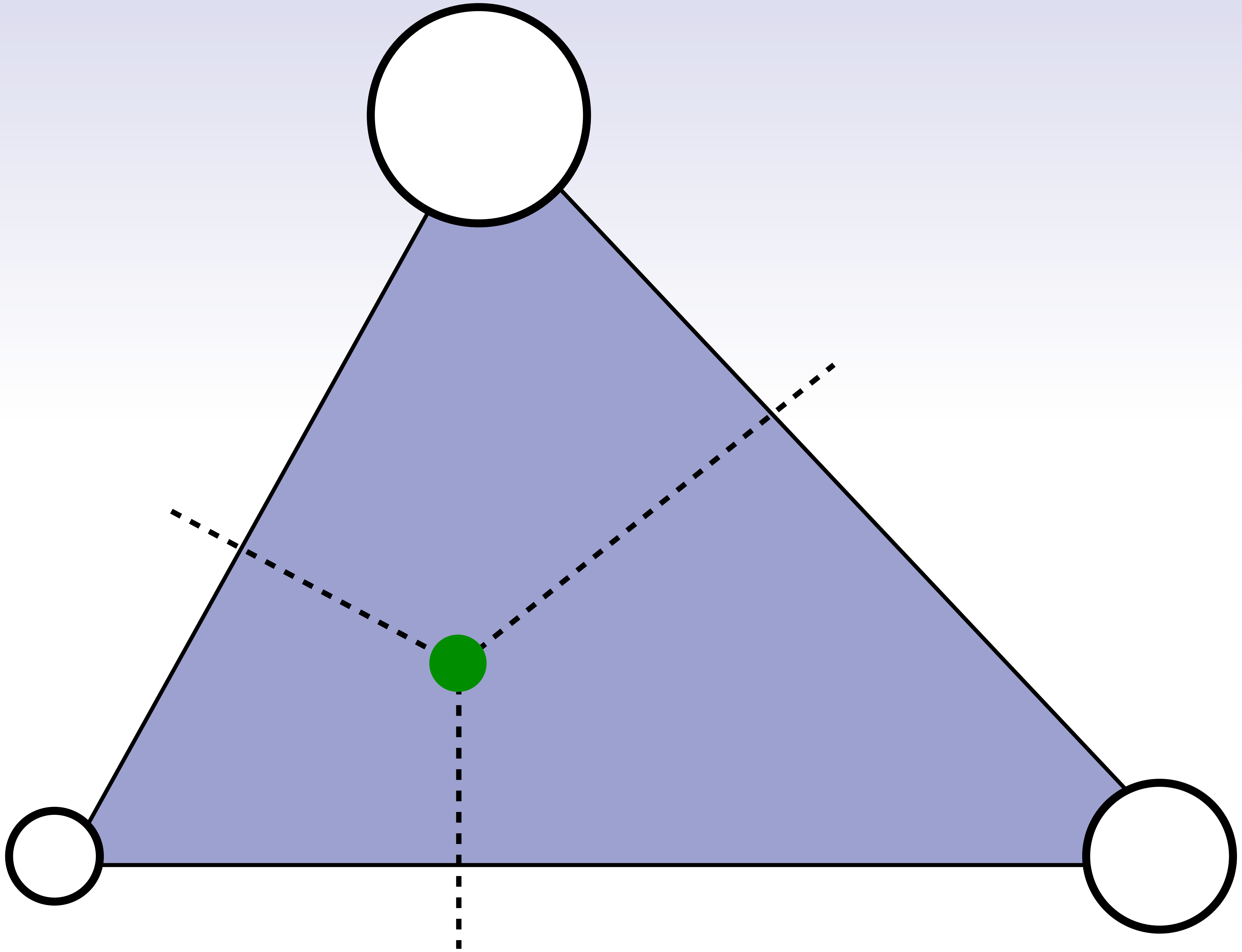
A structure is *self-supporting* when it stands in static equilibrium without external support. This idealization is fundamental when designing masonry structures which can withstand compression but are weak against tensile stresses.

Many techniques have been employed to design self-supporting structures, including the force network method [ODwyer 1999], the hanging chain model [Kilian and Ochsendorf 2005], and the thrust network [Fraternali 2010]. Recently, Block and Ochsendorf [2007; 2009] developed *thrust network analysis* which decouples the 3D force equilibrium of a self-supporting network into horizontal (xy)

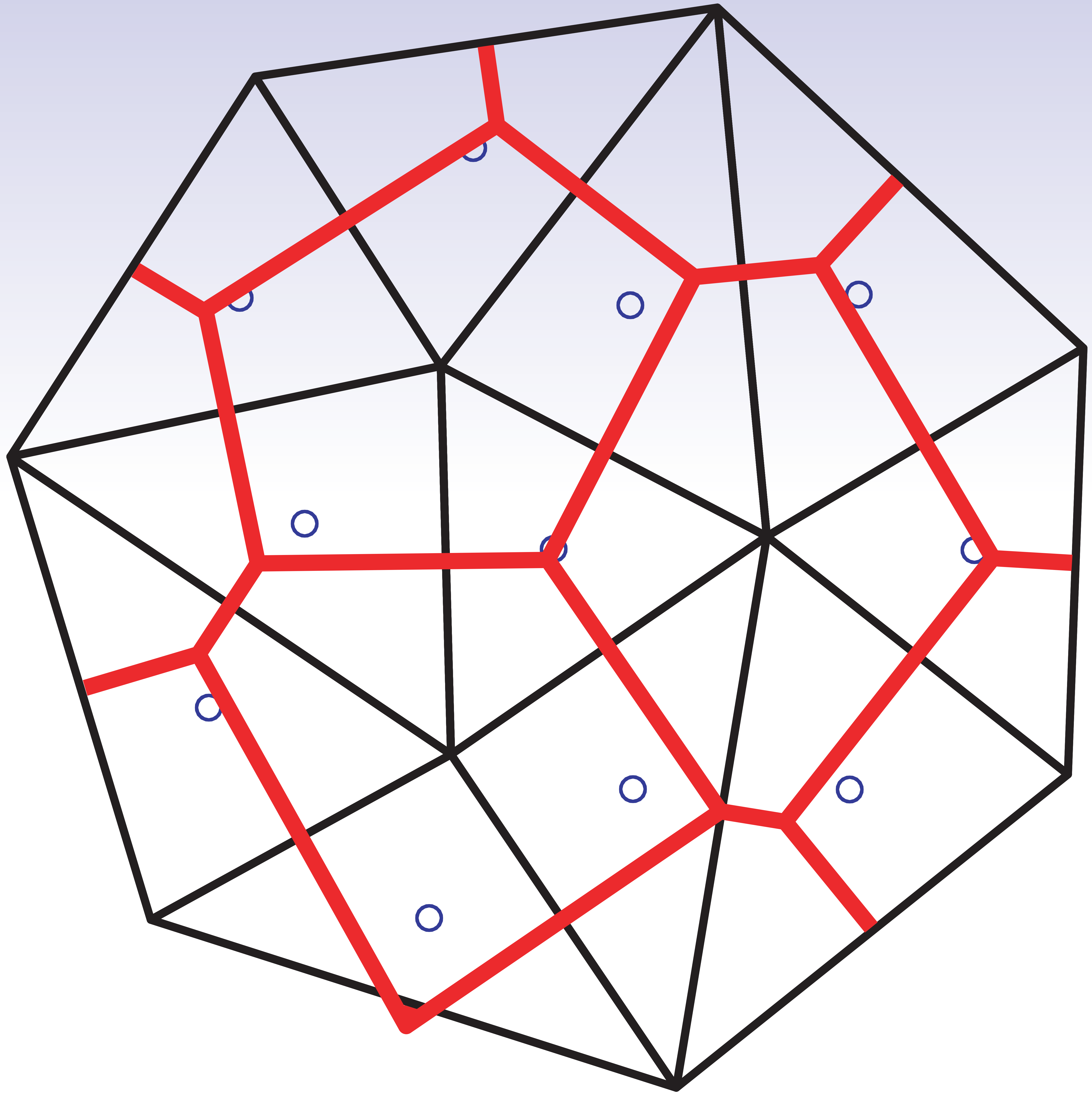
Weighted triangulations / Power diagrams



Weighted triangulations / Power diagrams

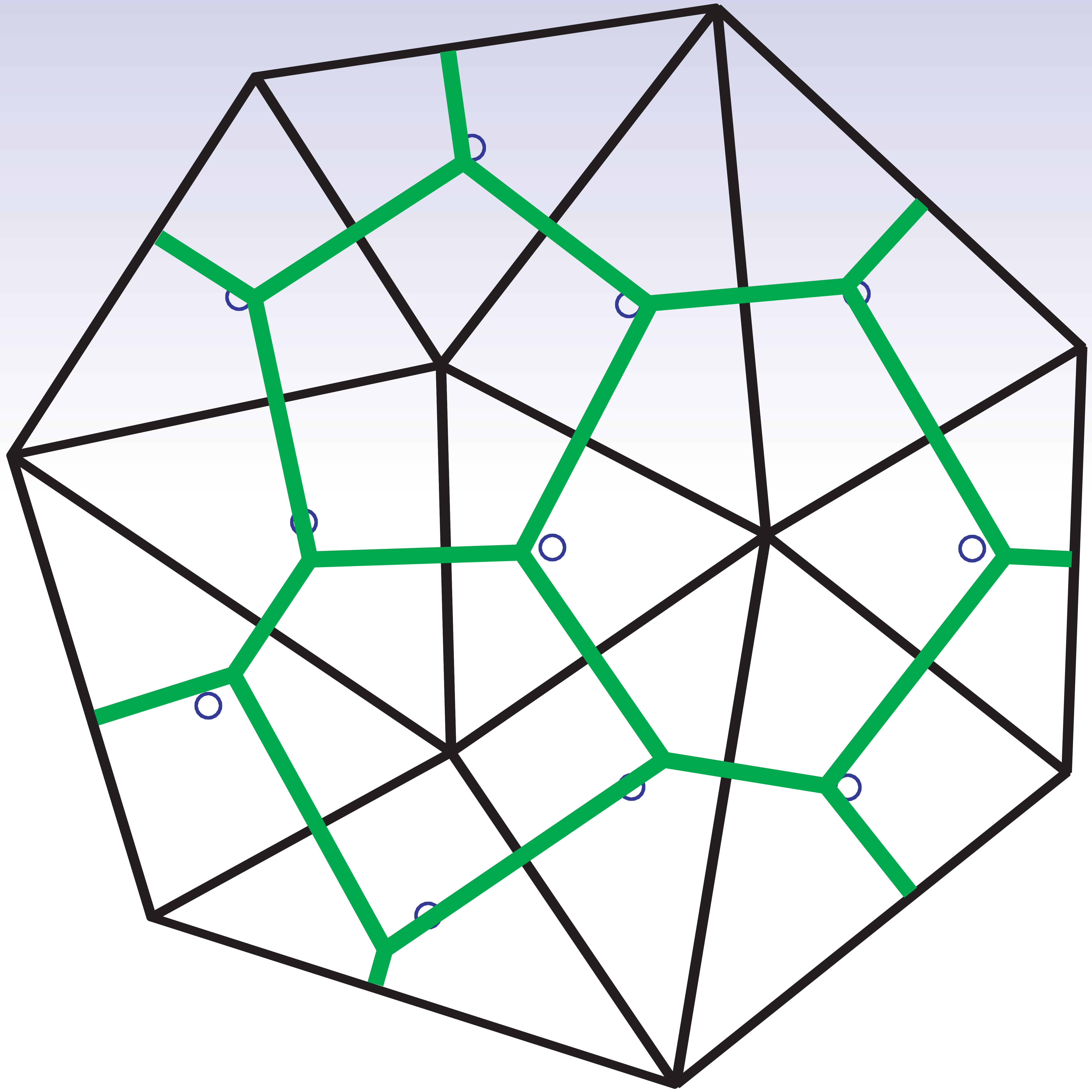


Well-centered meshes



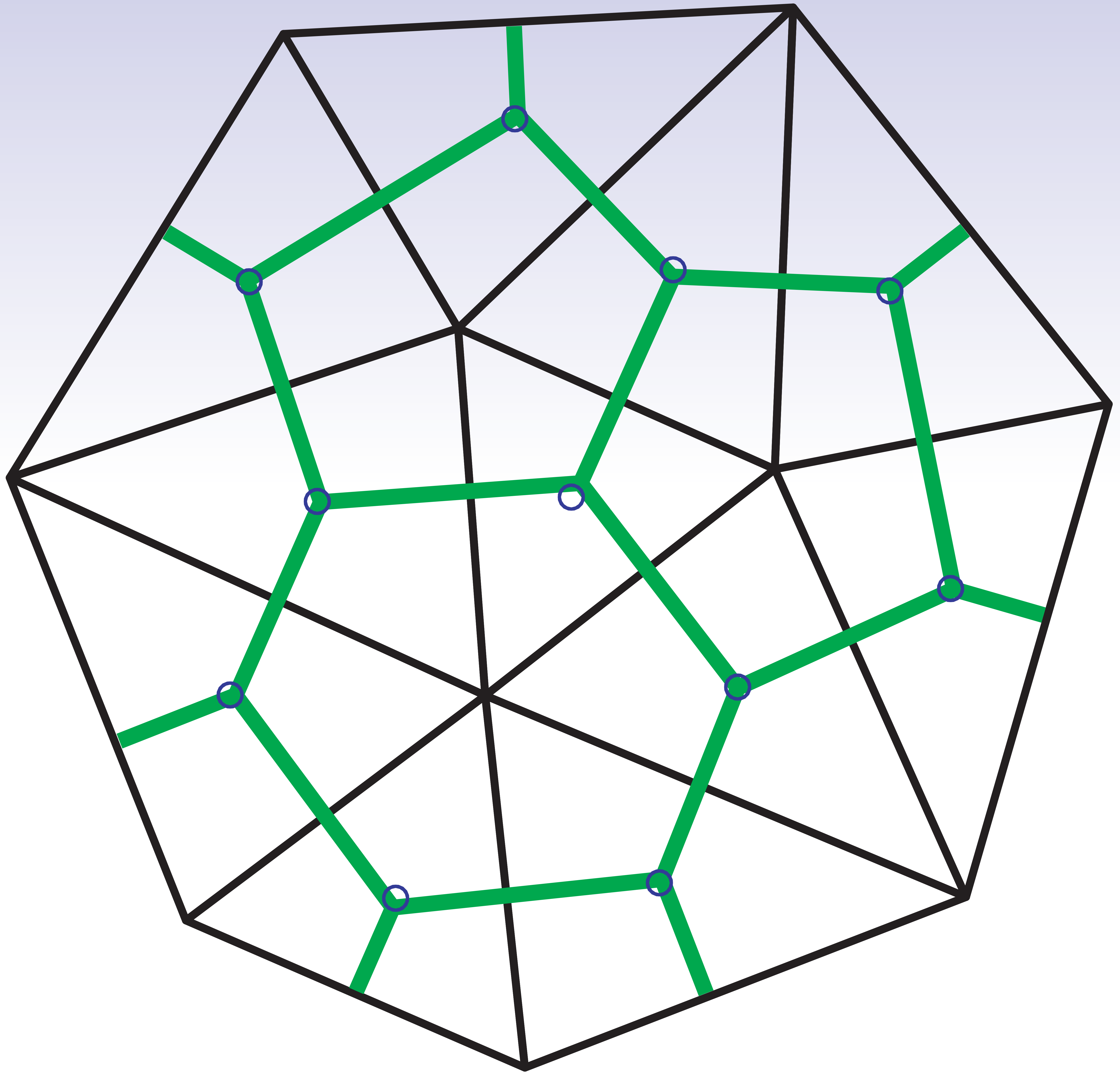
Mullen et al. 2011

Well-centered meshes



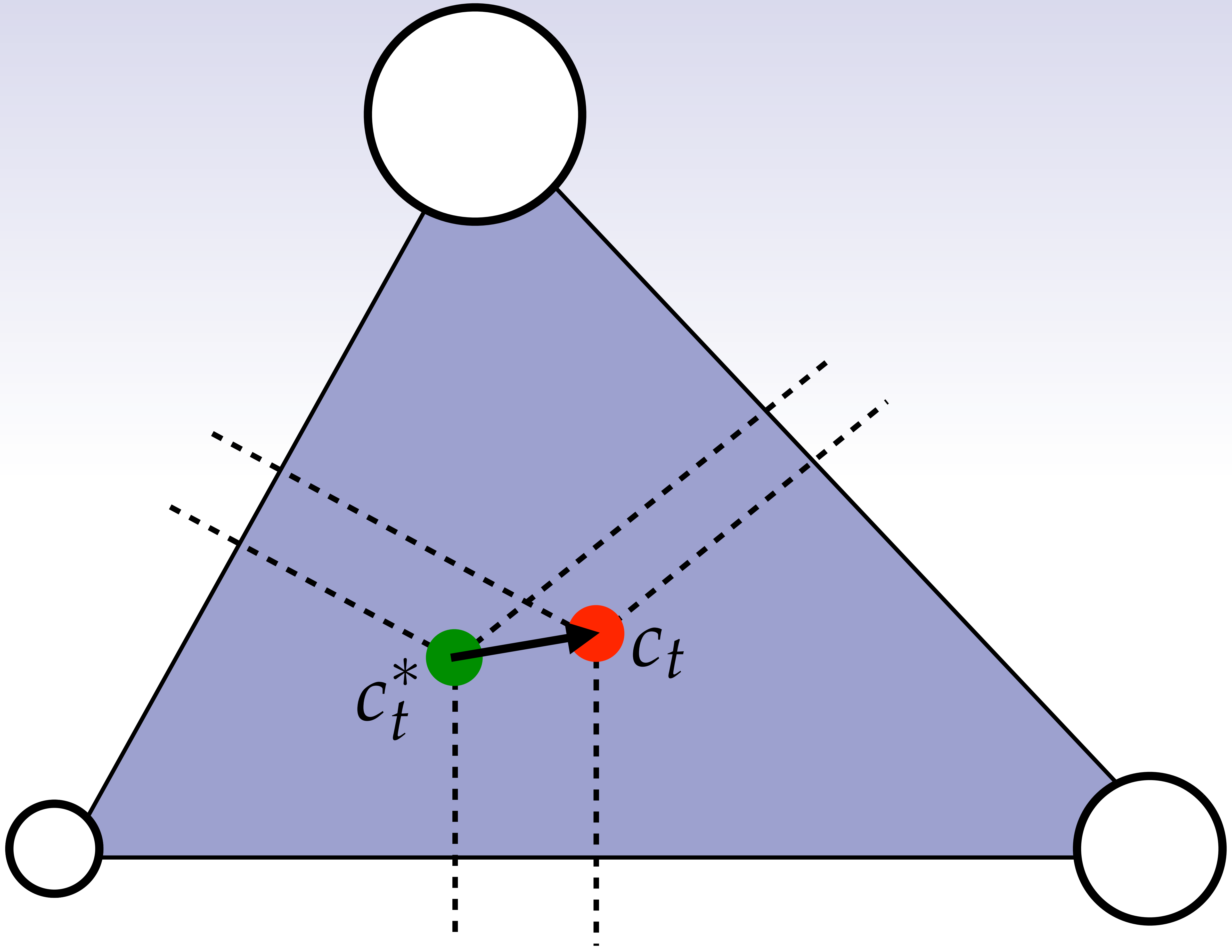
Mullen et al. 2011

Well-centered meshes



Mullen et al. 2011

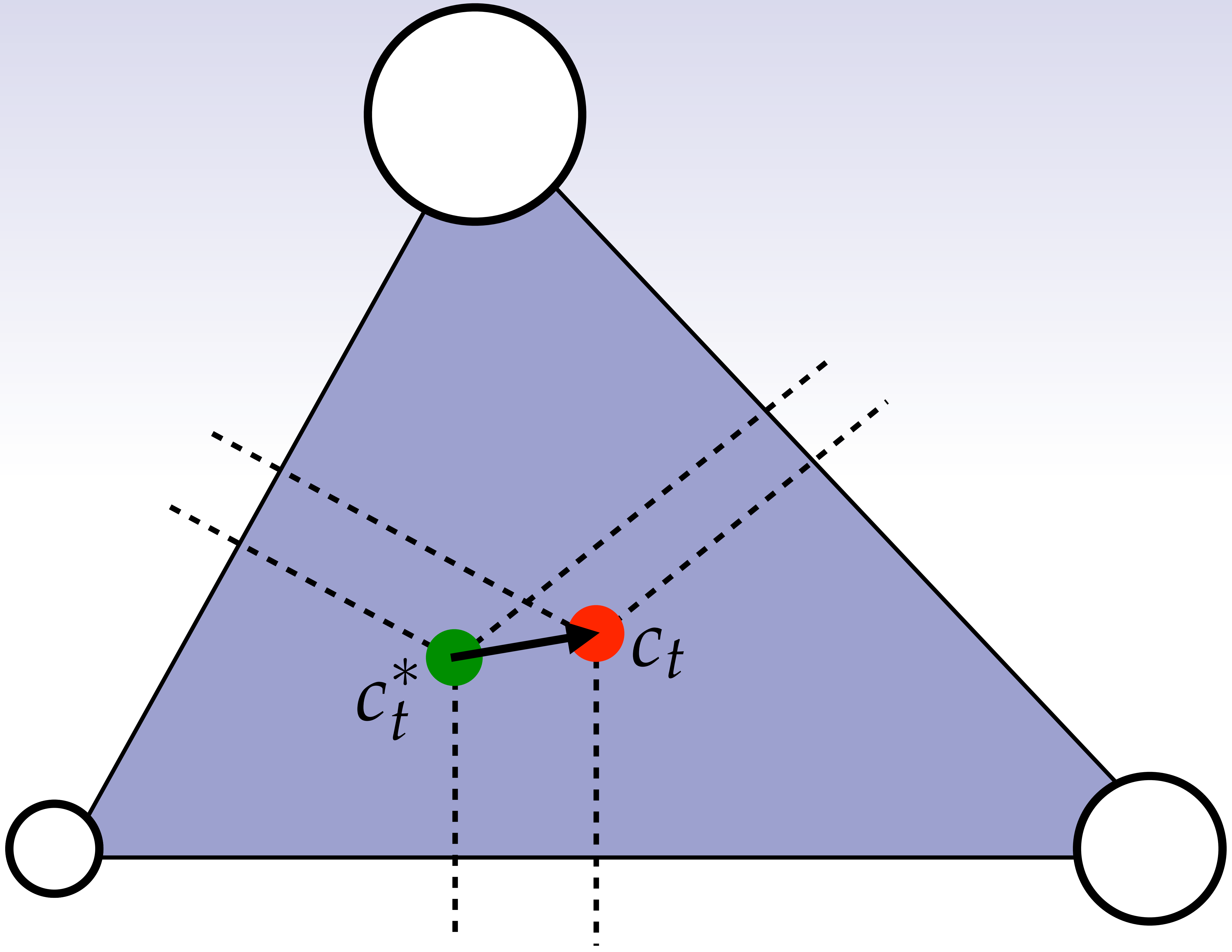
Improving dual



$$c_t - c_t^* = \frac{1}{2}\nabla w$$

de Goes et al. to appear

Improving dual

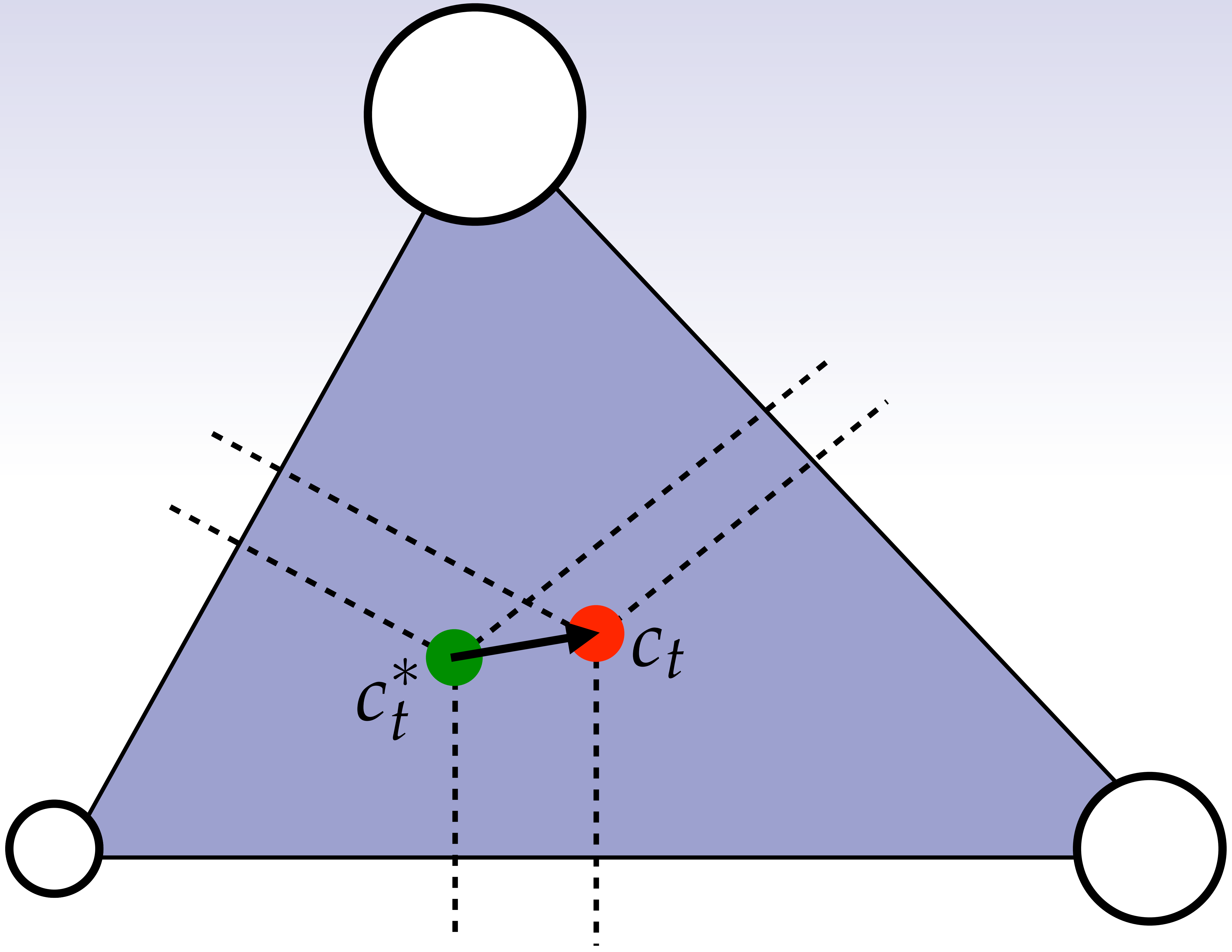


$$c_t - c_t^* = \frac{1}{2}\nabla w$$

$$\min_w \sum_i a_t \|(c_t - b_t) - \tfrac{1}{2}\nabla w\|^2$$

de Goes et al. to appear

Improving dual



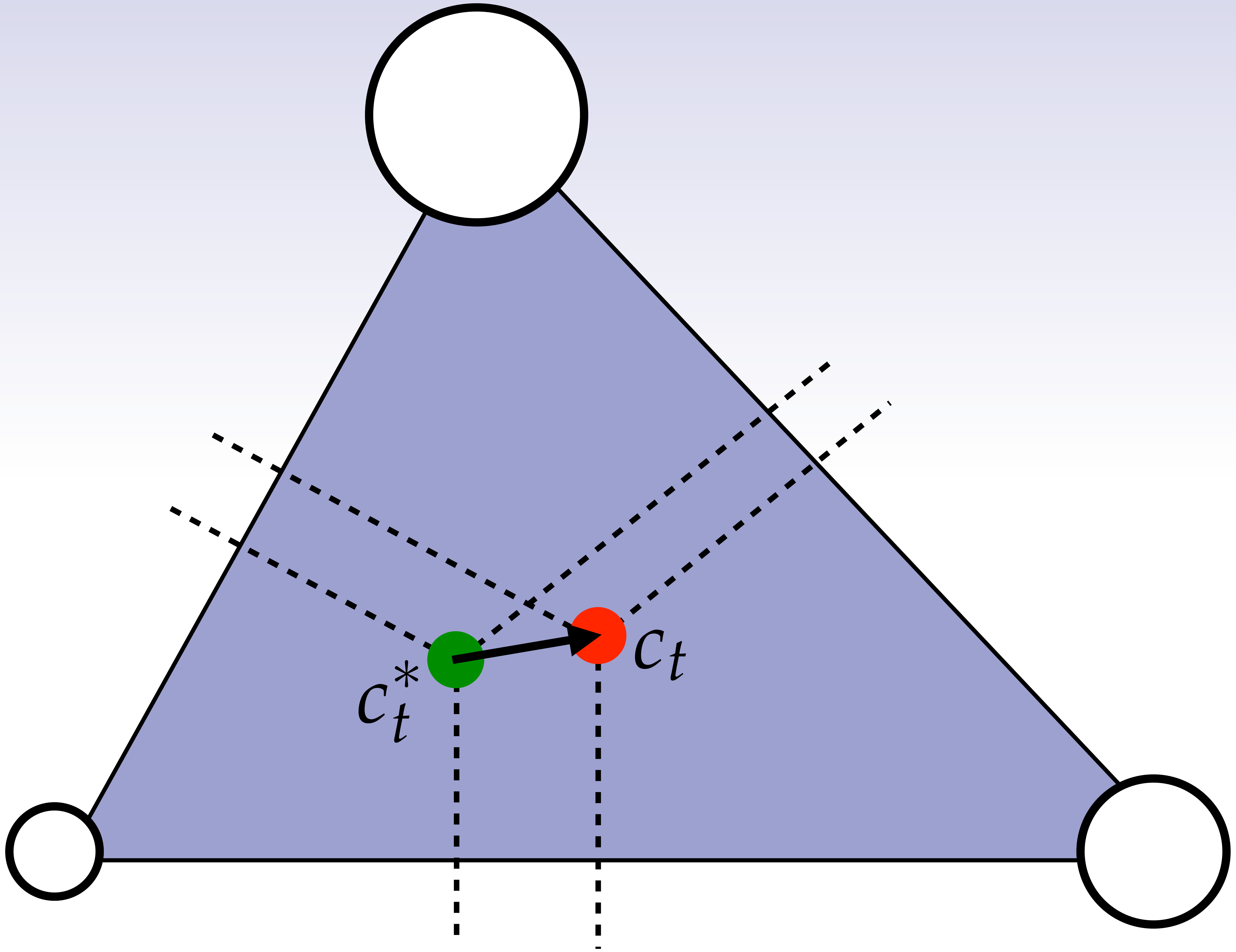
$$c_t - c_t^* = \frac{1}{2}\nabla w$$

$$\min_w \sum_i a_t \|(c_t - b_t) - \tfrac{1}{2}\nabla w\|^2$$

$$\Delta w = \nabla \cdot (c - b)$$

de Goes et al. to appear

Improving dual



$$c_t - c_t^* = \frac{1}{2}\nabla w$$

$$\min_w \sum_i a_t \|(c_t - b_t) - \tfrac{1}{2}\nabla w\|^2$$

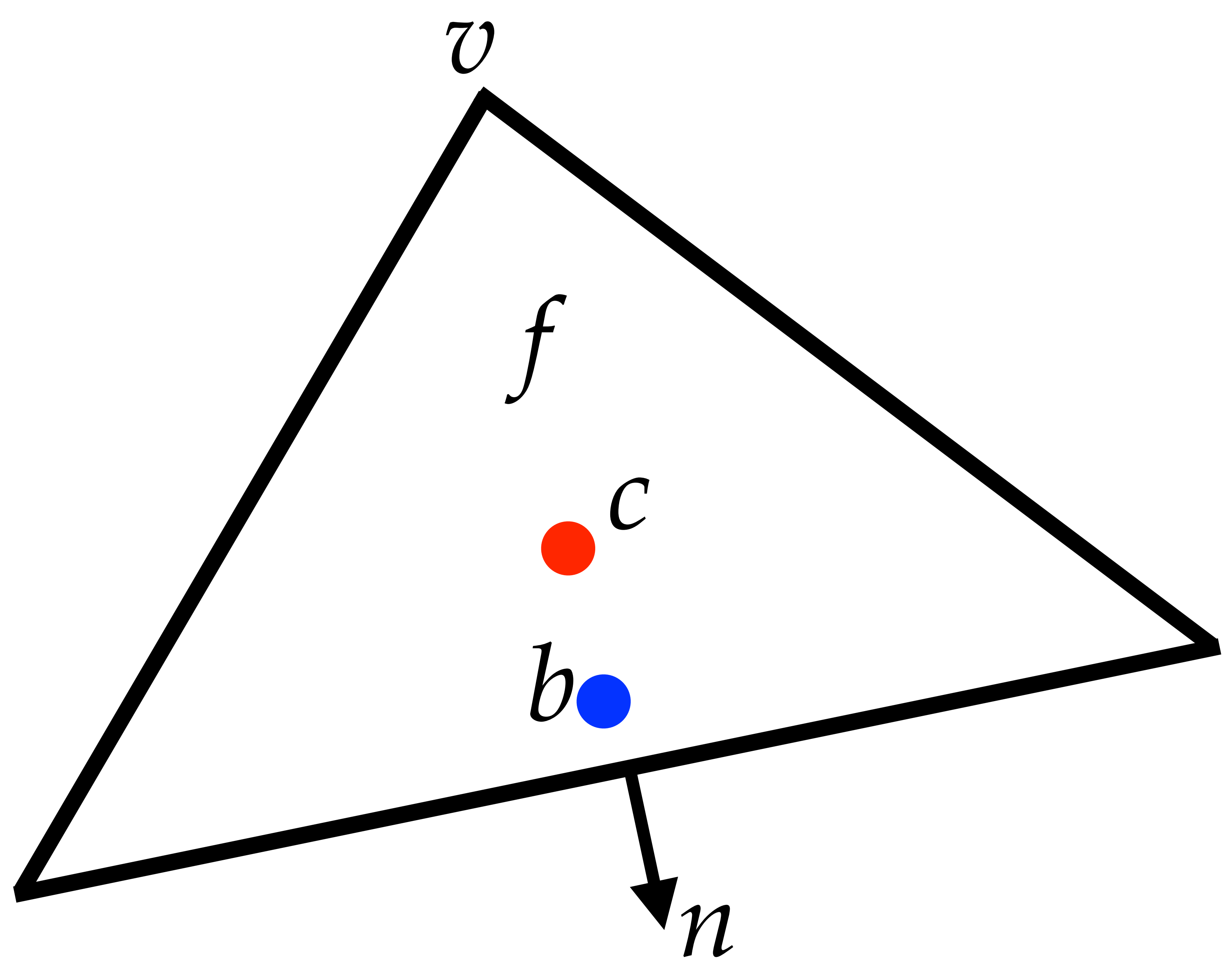
$$\Delta w = \nabla \cdot (c - b)$$

$$\min_x \sum_i a_t \|(c_t - b_t) - \tfrac{1}{2}\nabla w\|^2$$

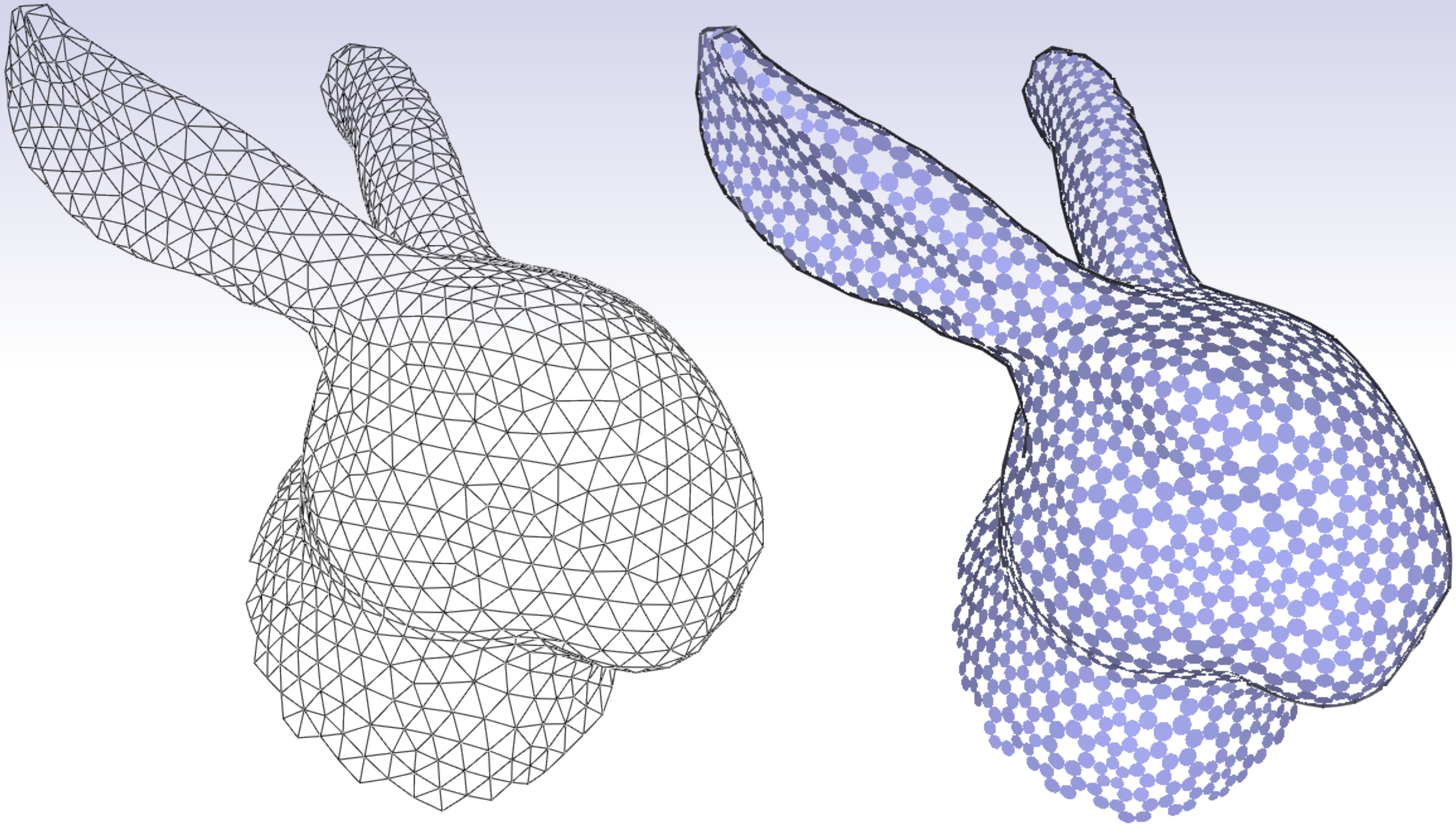
de Goes et al. to appear

Pseudo-code

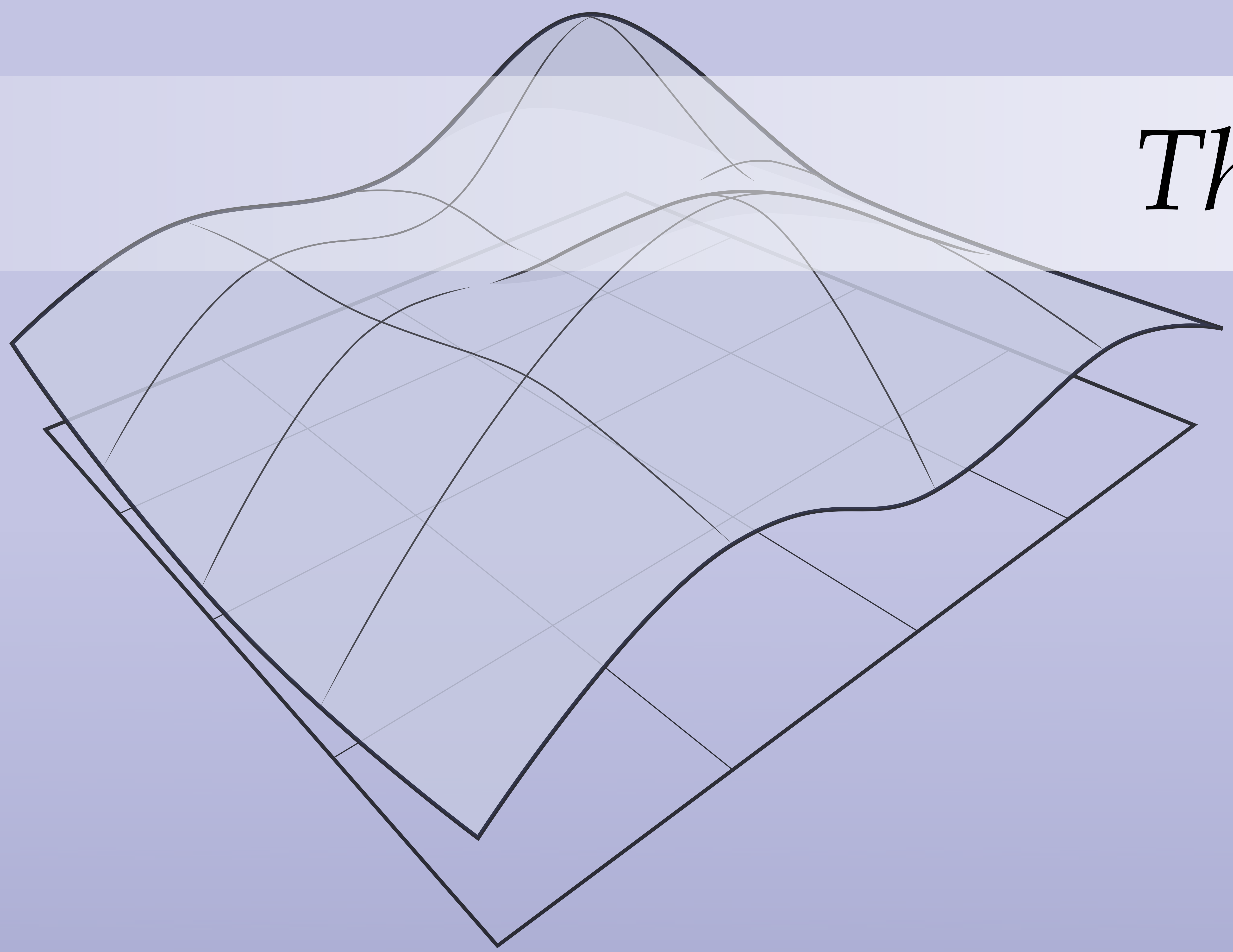
```
// build matrix
SparseMatrix<Real> L = d0.transpose() * star1 * d0;

// build rhs
DenseMatrix<Real> rhs(#Vertices);
foreach vertex v:
   foreach face f in OneRing(v):
      rhs(v) += dot( n, b-c );

// solve for w
solve(L, w, rhs);

// quasi-newton solver for x
optimize_vertices()
```



Example



Thanks!



DIGITAL GEOMETRY PROCESSING WITH DISCRETE EXTERIOR CALCULUS

Keenan Crane • Fernando de Goes • Mathieu Desbrun • Peter Schröder