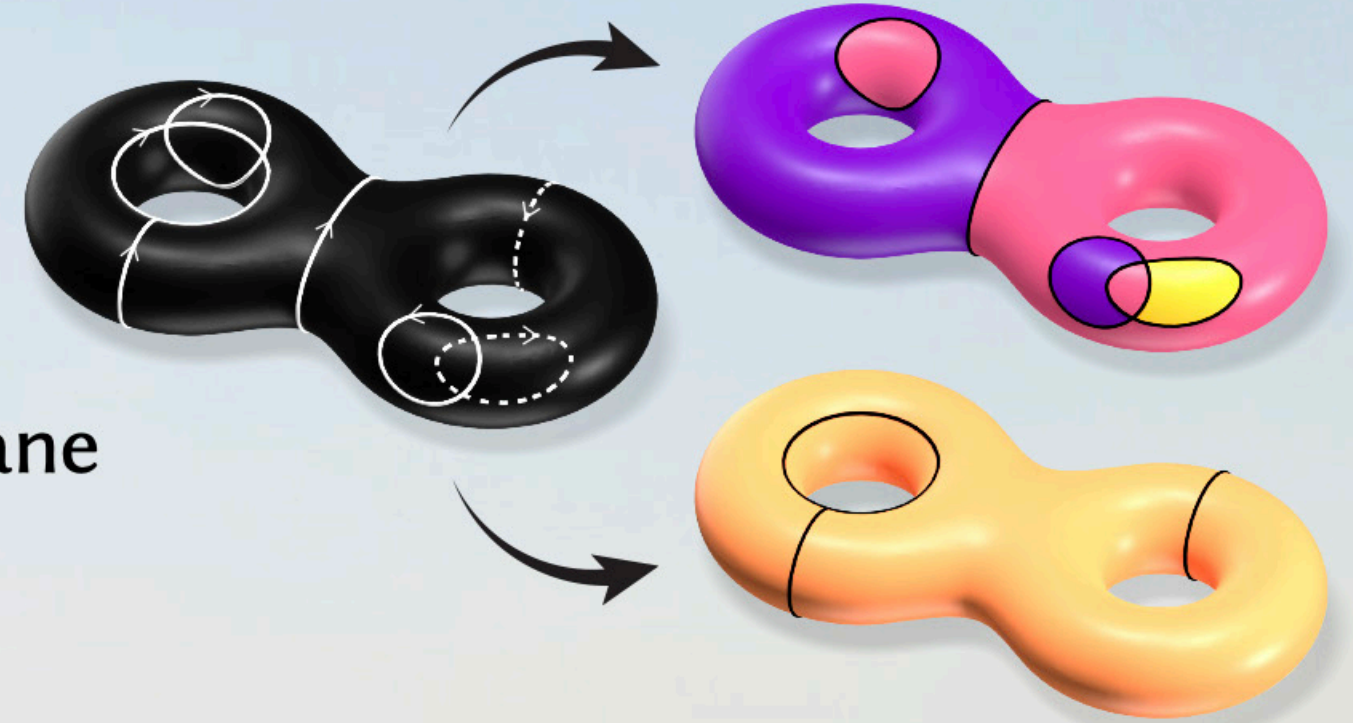


WINDING NUMBERS ON DISCRETE SURFACES

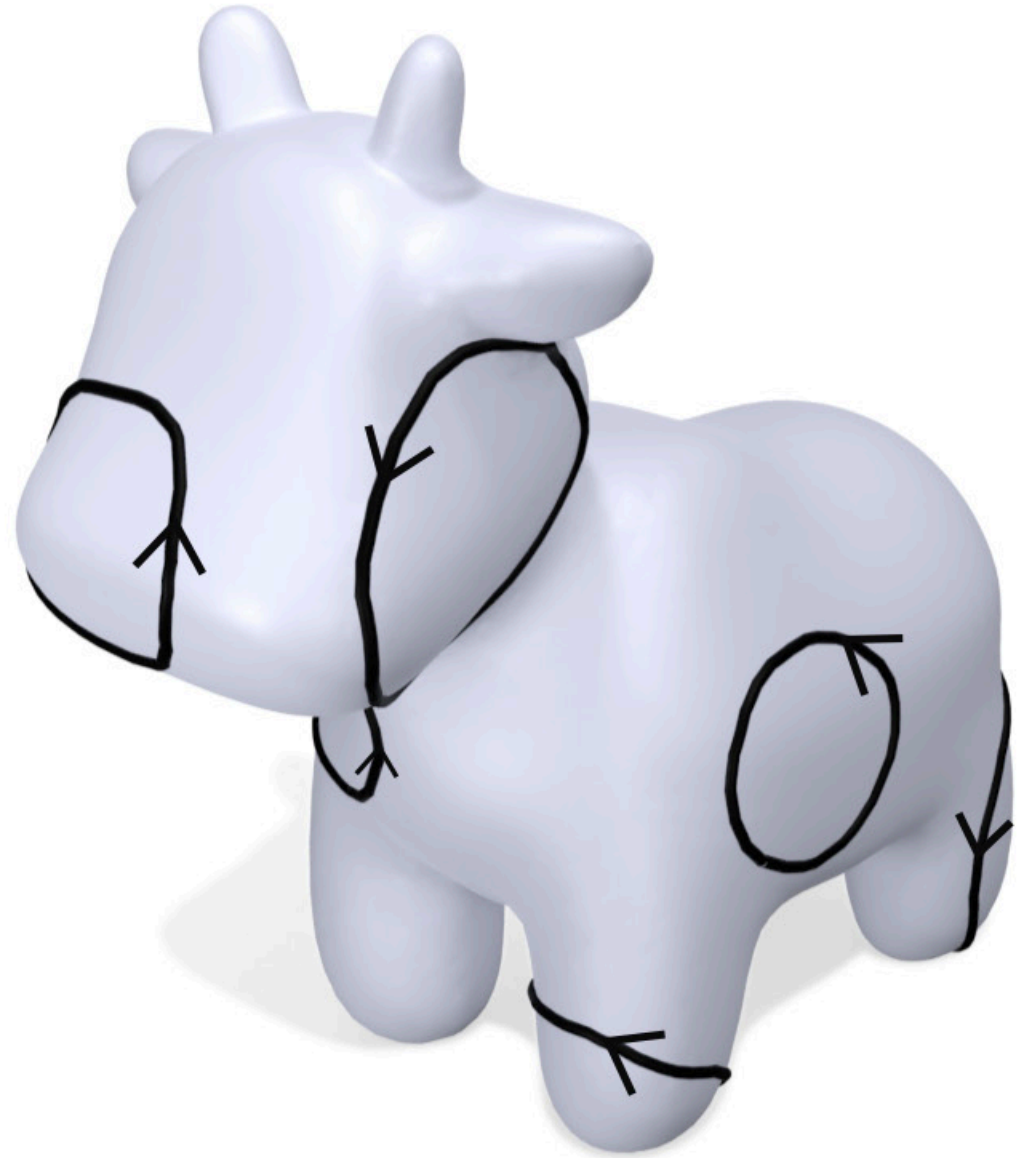
Nicole Feng, Mark Gillespie, Keenan Crane

Carnegie Mellon University

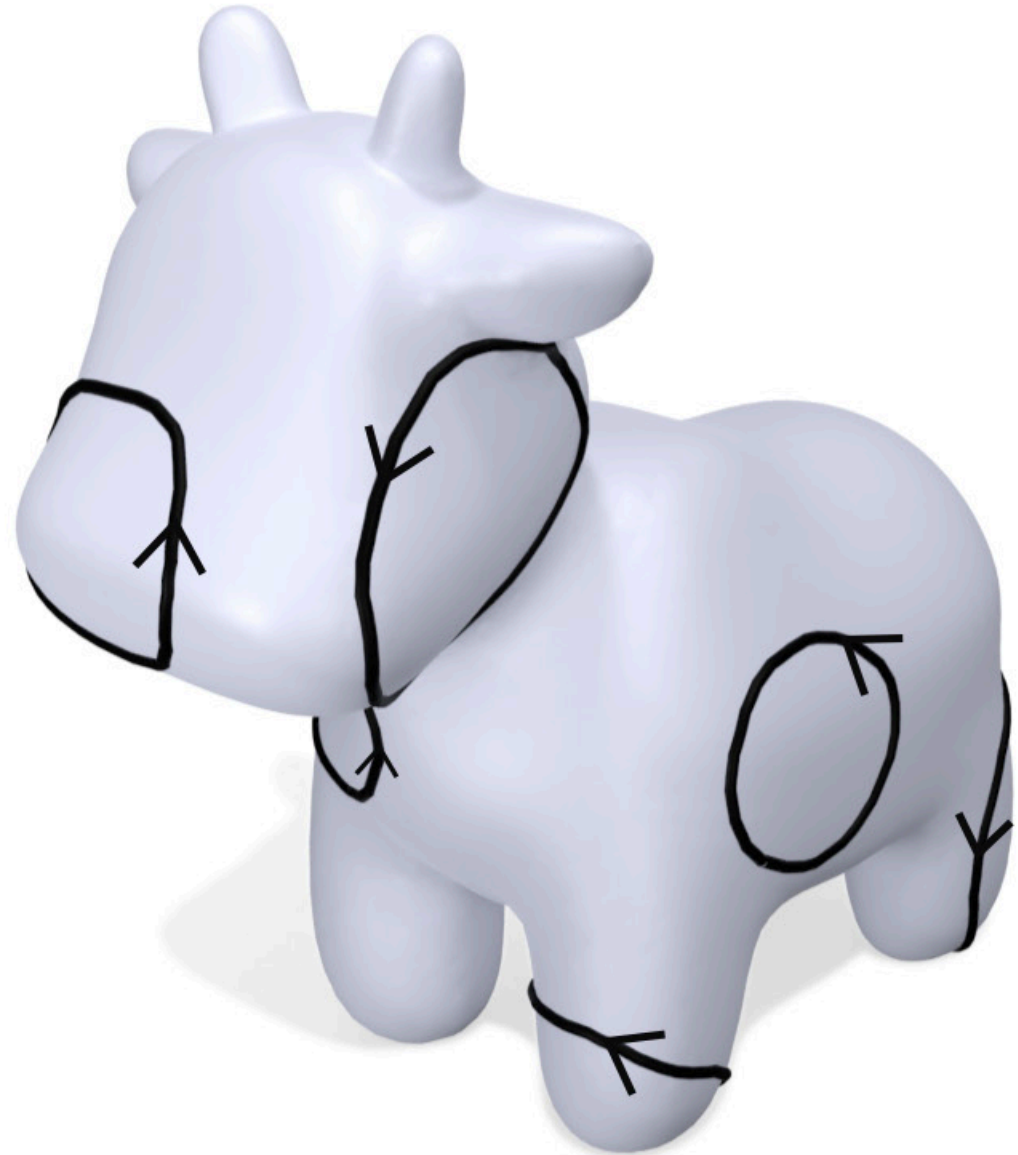


[date]

Problem



Problem



Not all curves bound regions!

Not all curves bound regions!

Fence bounds a region:



Not all curves bound regions!

Fence bounds a region:



Fence doesn't bound a region:

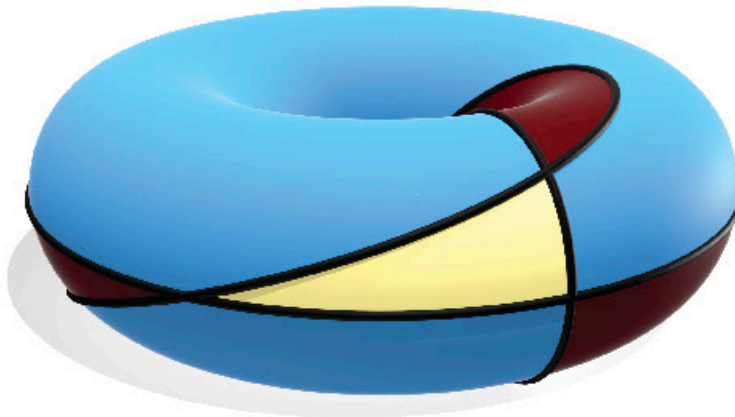
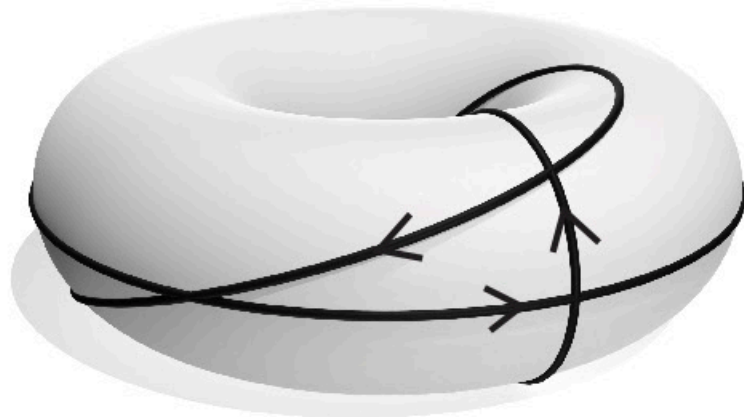


Not all curves bound regions!

Fence bounds a region:

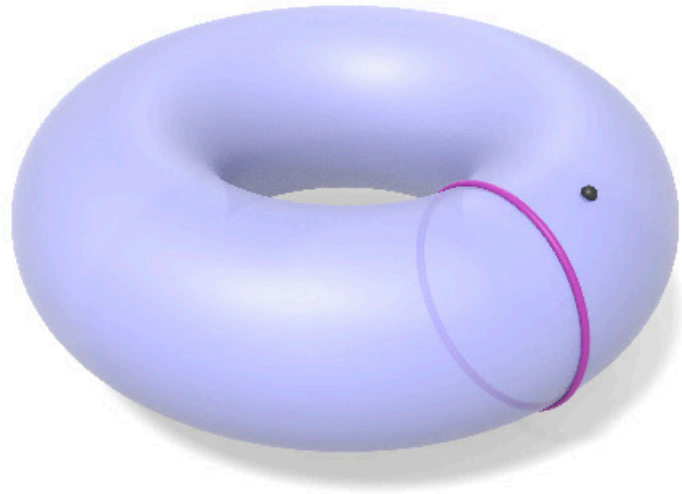


Fence doesn't bound a region:

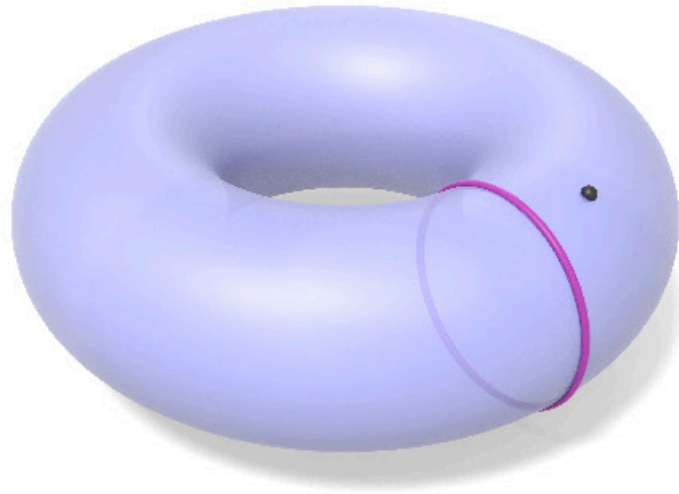


Not as simple as
classifying loops

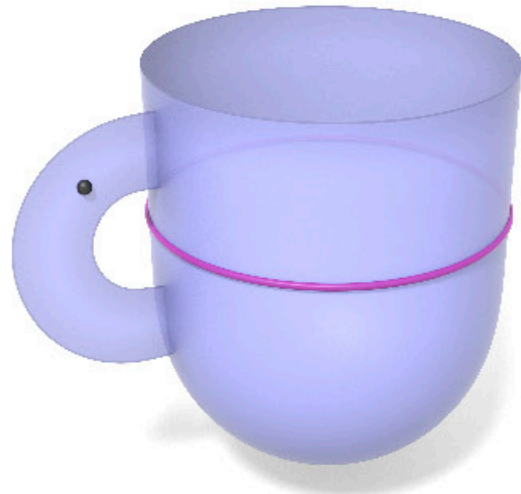
Is the point inside or outside the curve?



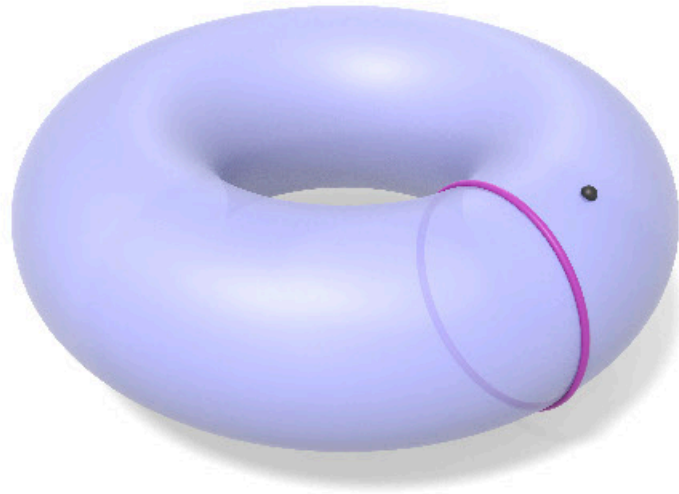
Is the point inside or outside the curve?



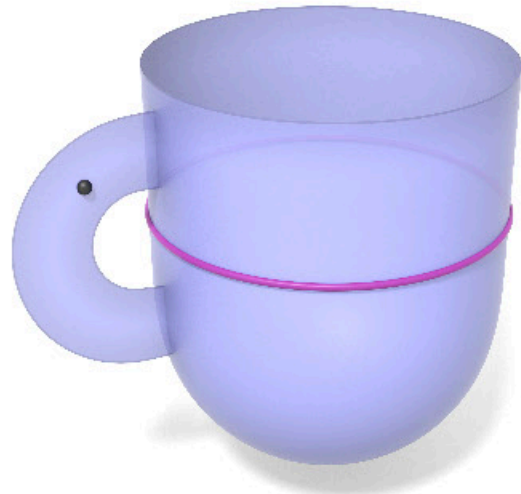
non-manifold



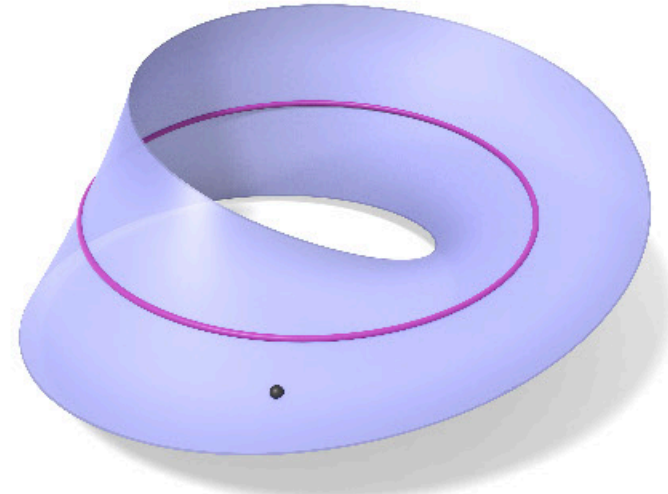
Is the point inside or outside the curve?



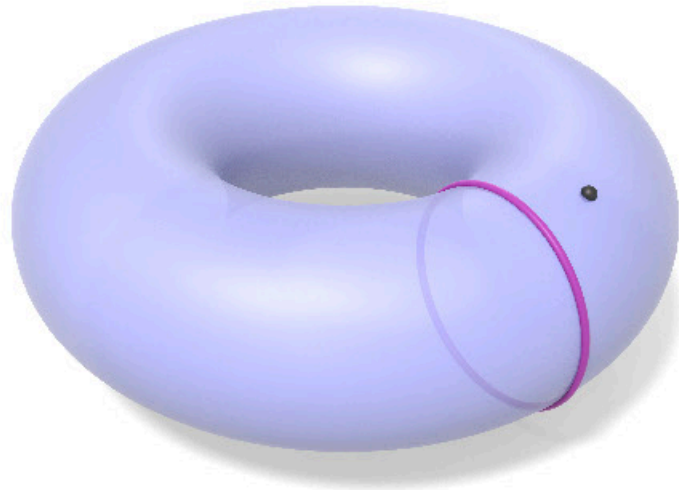
non-manifold



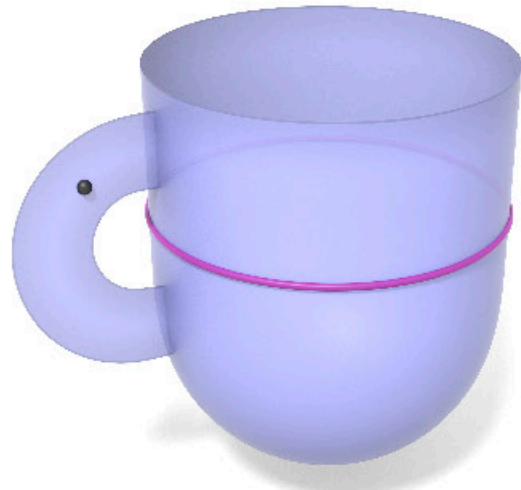
non-orientable



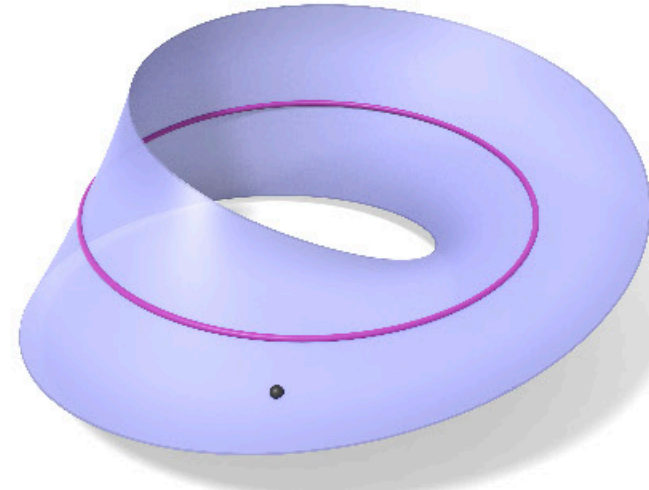
Is the point inside or outside the curve?



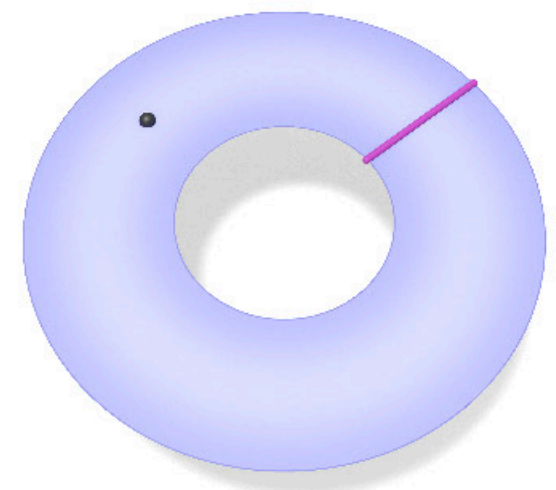
non-manifold



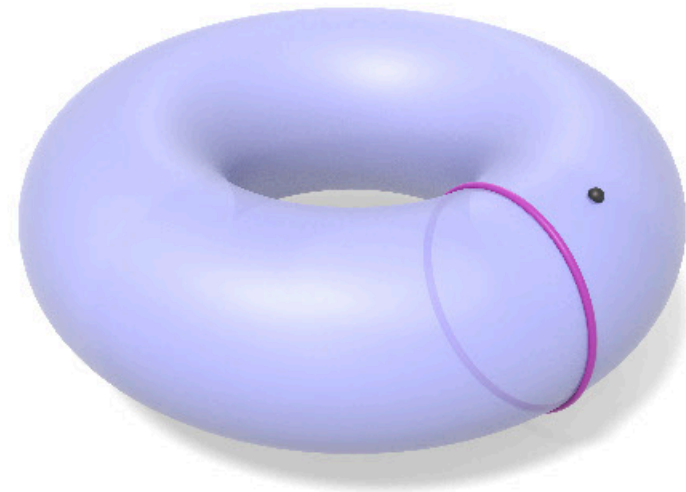
non-orientable



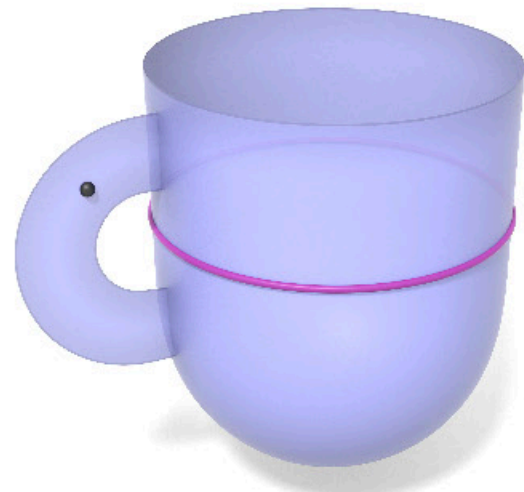
with boundary



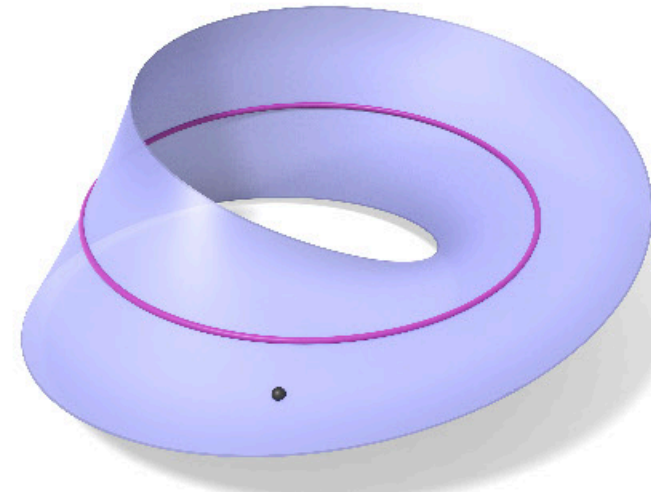
Is the point inside or outside the curve?



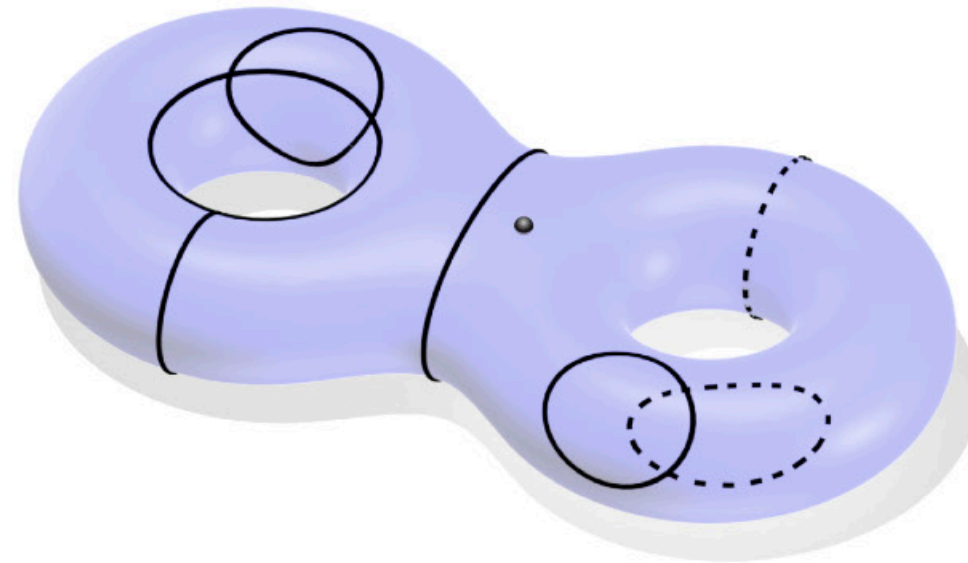
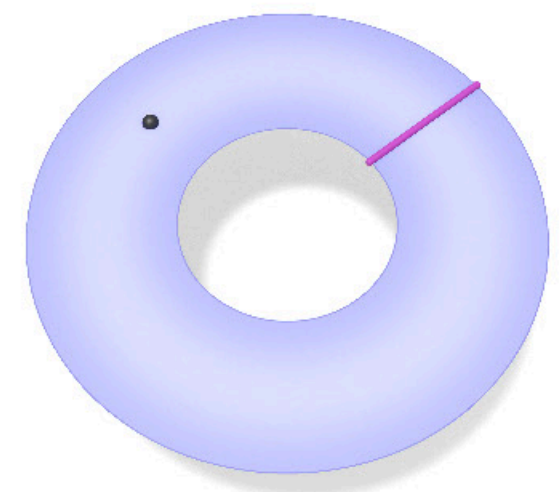
non-manifold



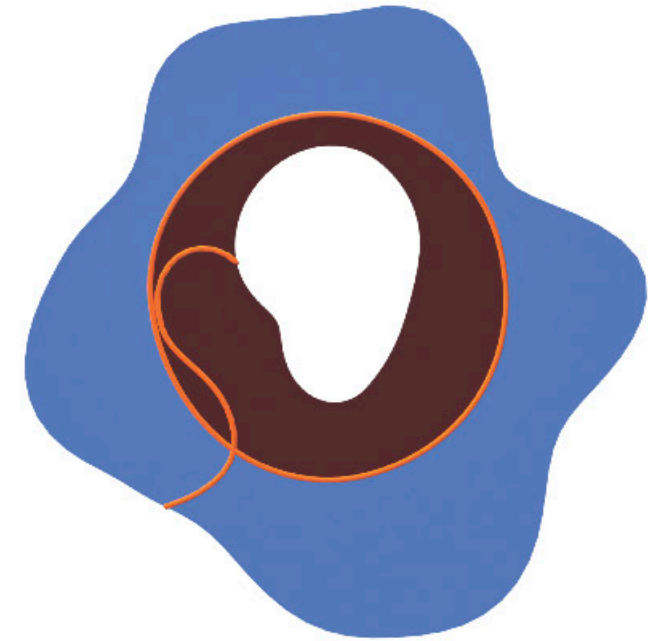
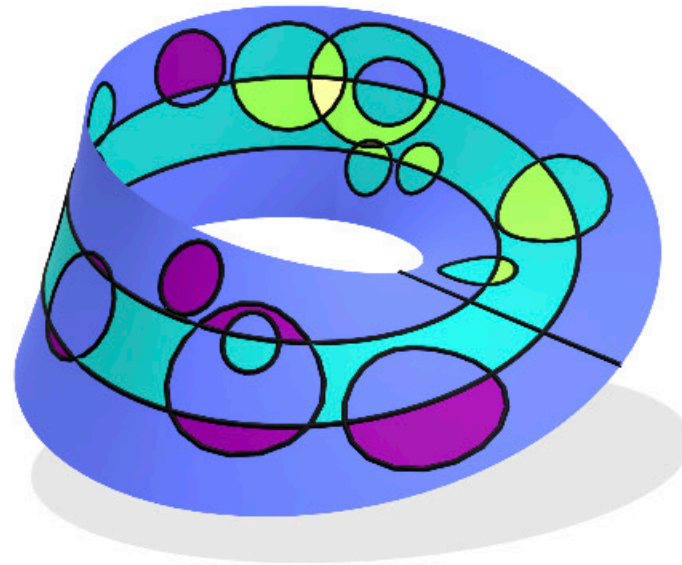
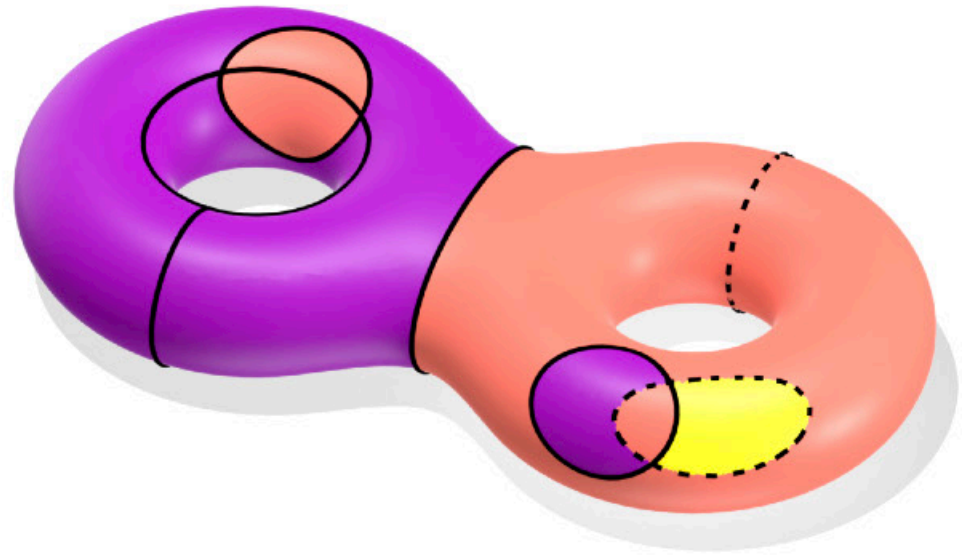
non-orientable



with boundary

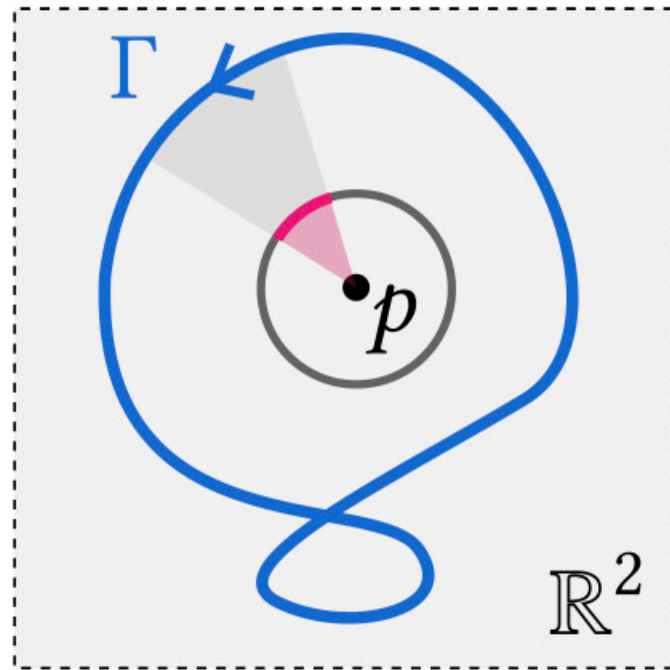


Surface Winding Numbers (SWN)

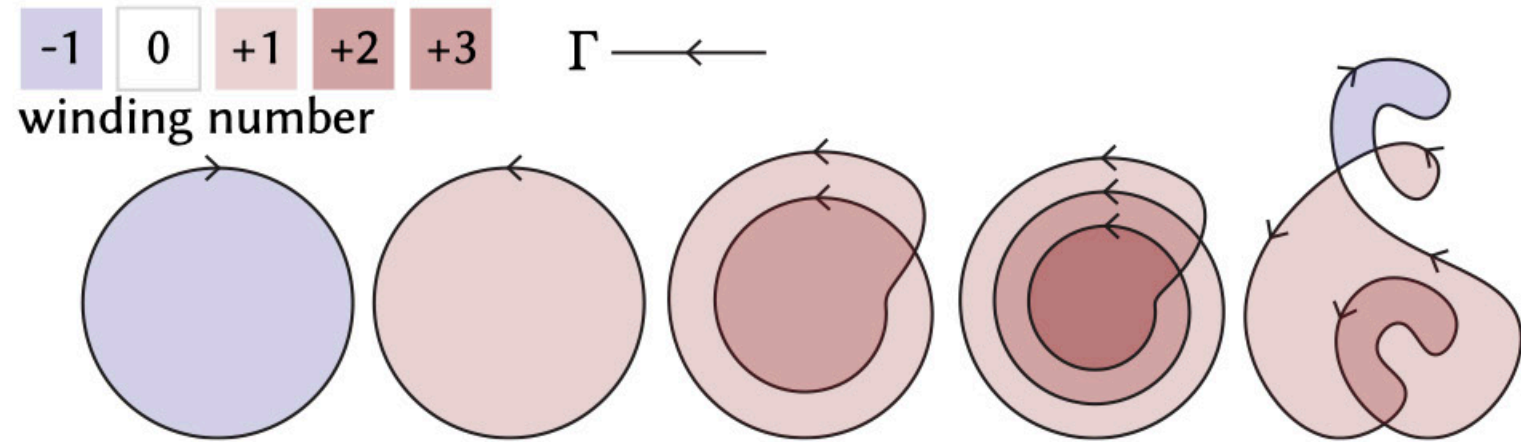
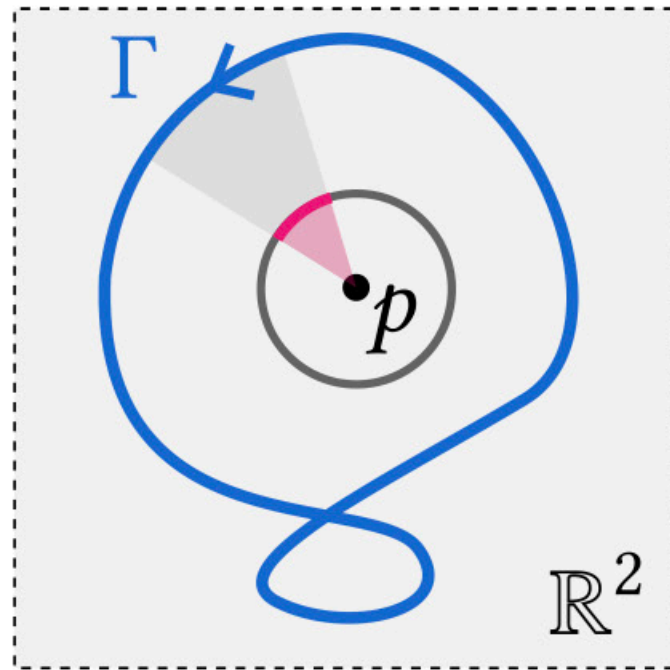


Winding number (solid angle)

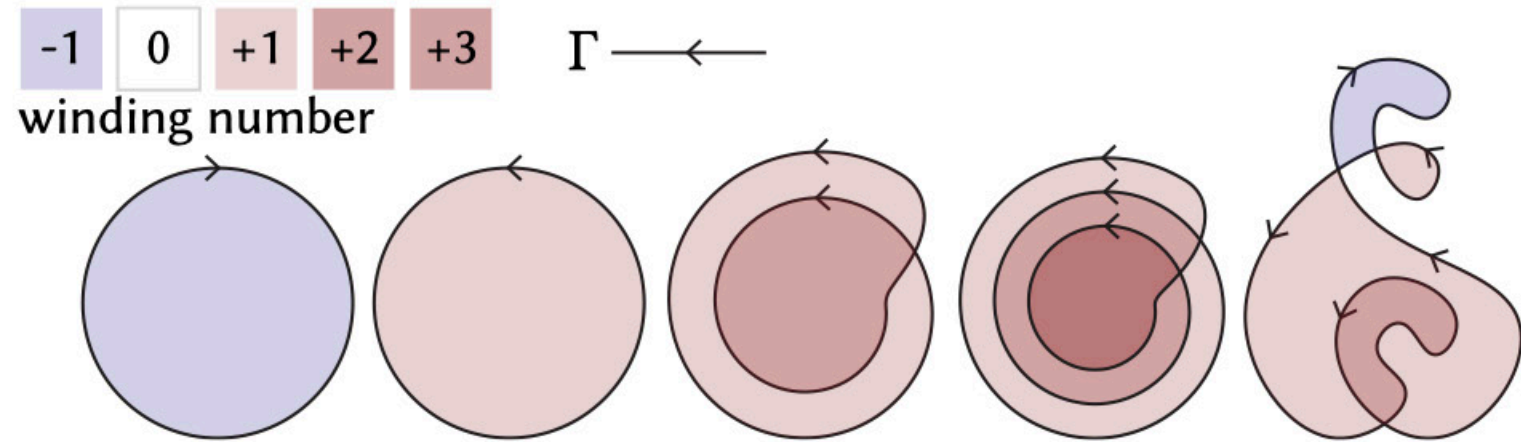
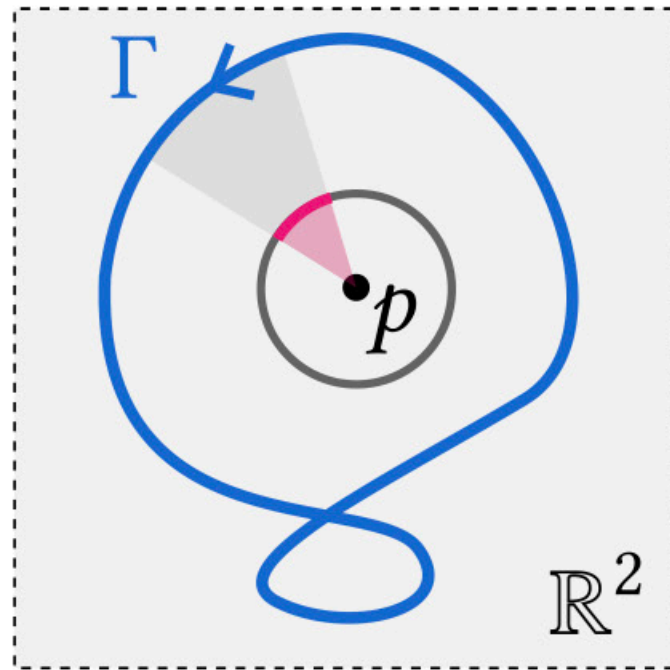
Winding number (solid angle)



Winding number (solid angle)



Winding number (solid angle)

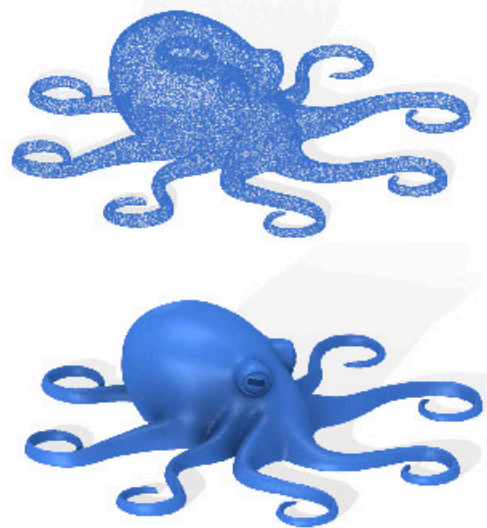


Complex analysis, differential geometry, topology, electromagnetism...

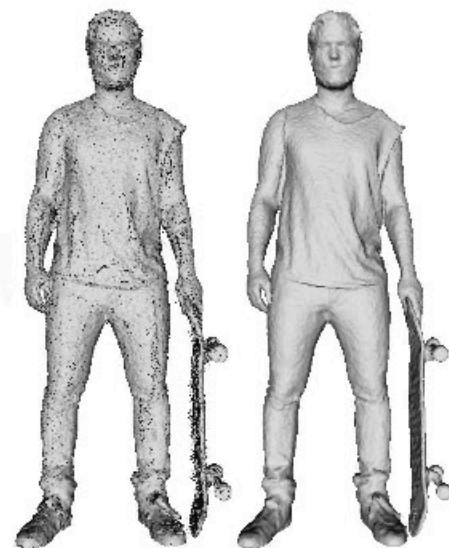
Winding numbers have succeeded before!

Winding numbers have succeeded before!

surface reconstruction



[Barill et al. 2018]



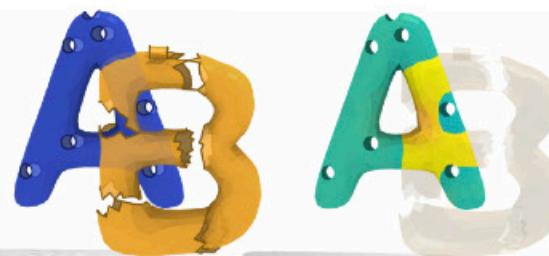
[Collet et al. 2015]



[Zhou et al. 2016]



mesh booleans



[Barill et al. 2018]

iterative normal estimation

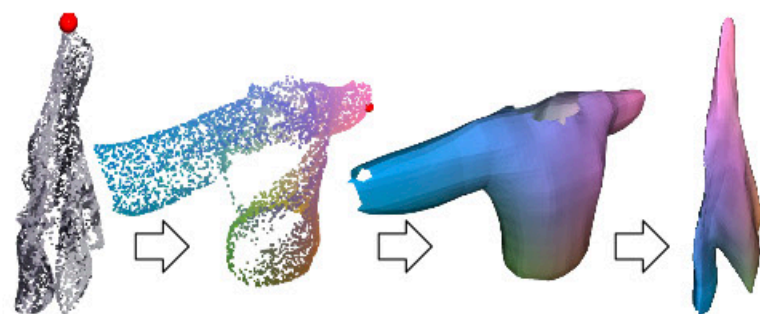


[Xu et al. 2023]

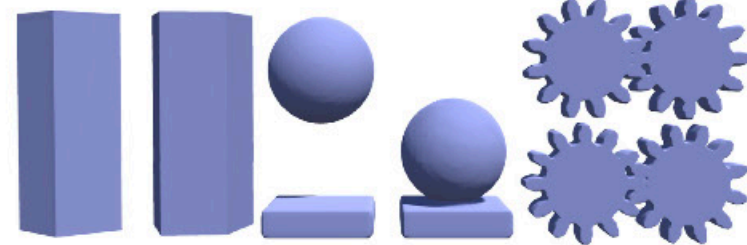


[Hou et al. 2022]

geometric preprocessing

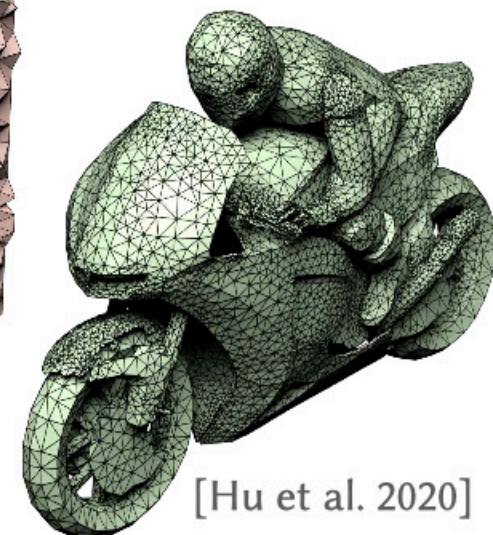
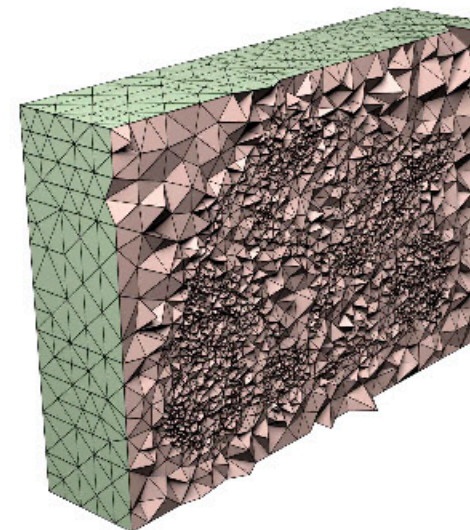


[Chi, Song 2021]



[Dvořák et al. 2021]

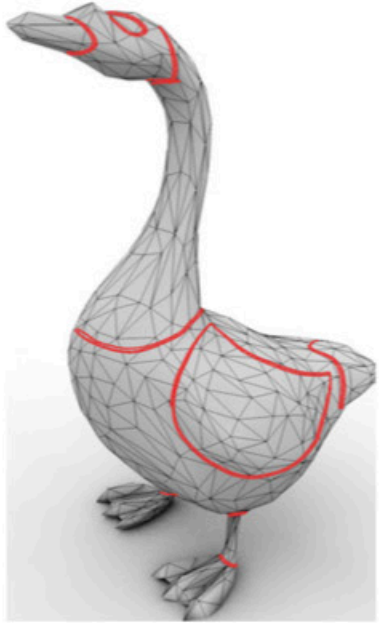
tet-meshing



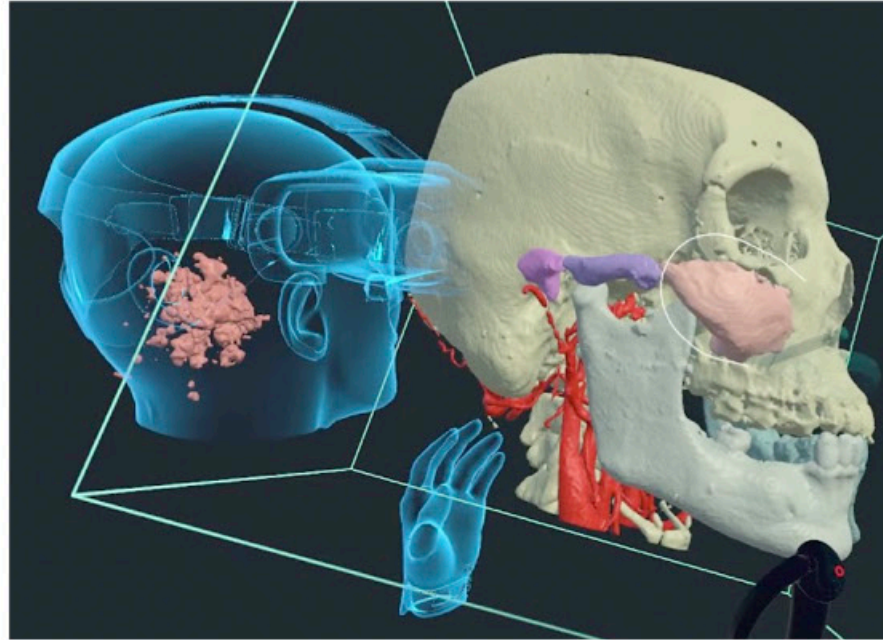
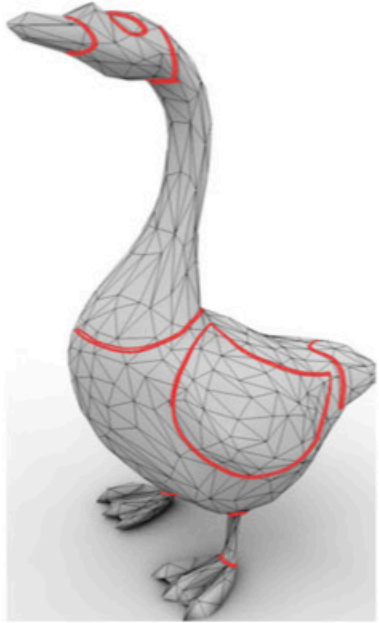
[Hu et al. 2020]

Our motivation

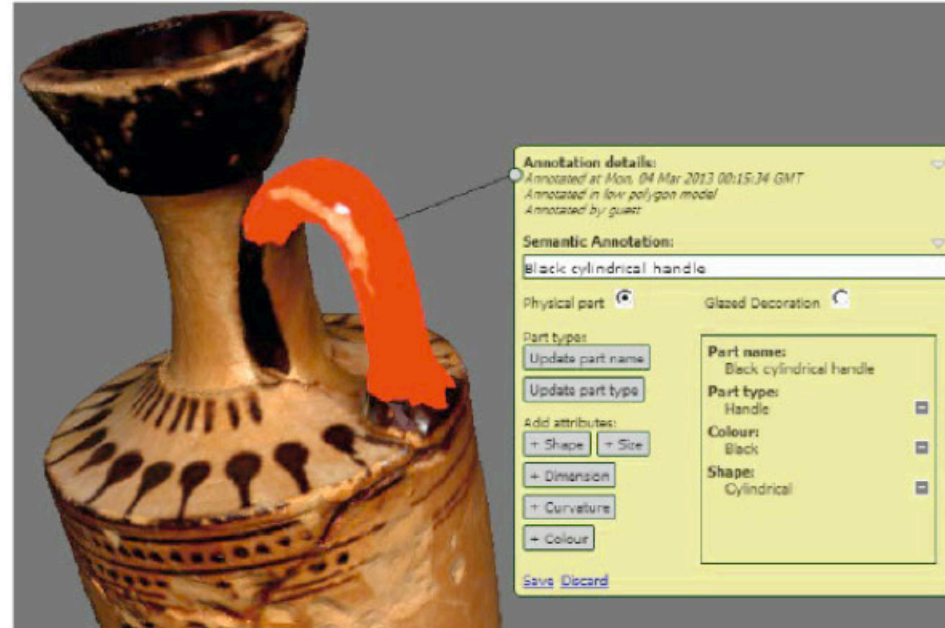
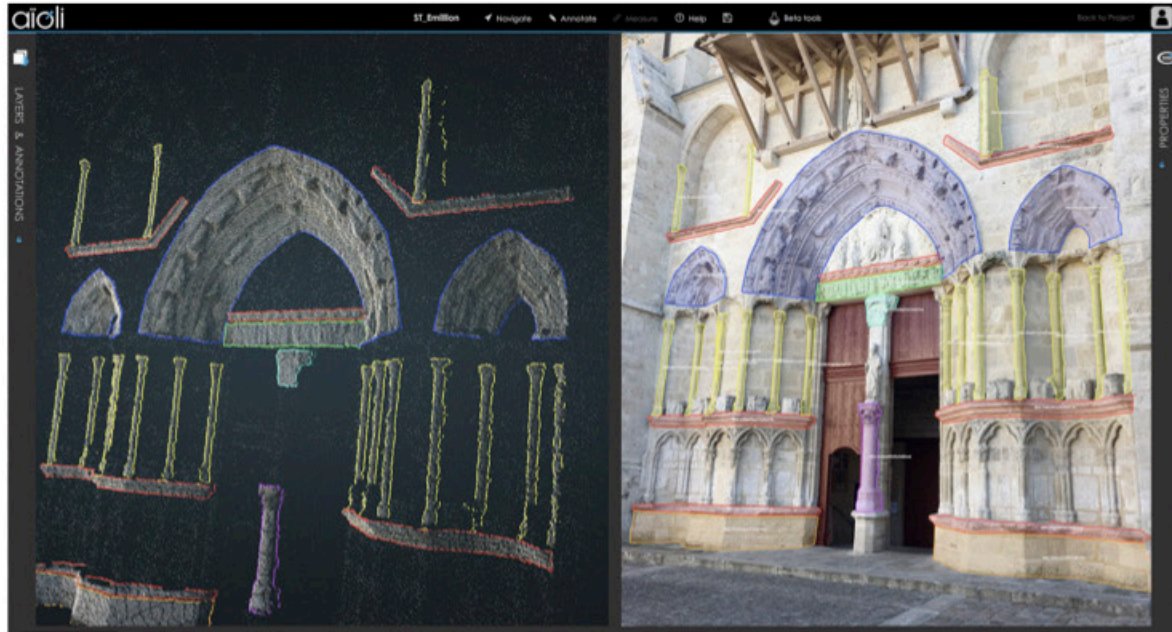
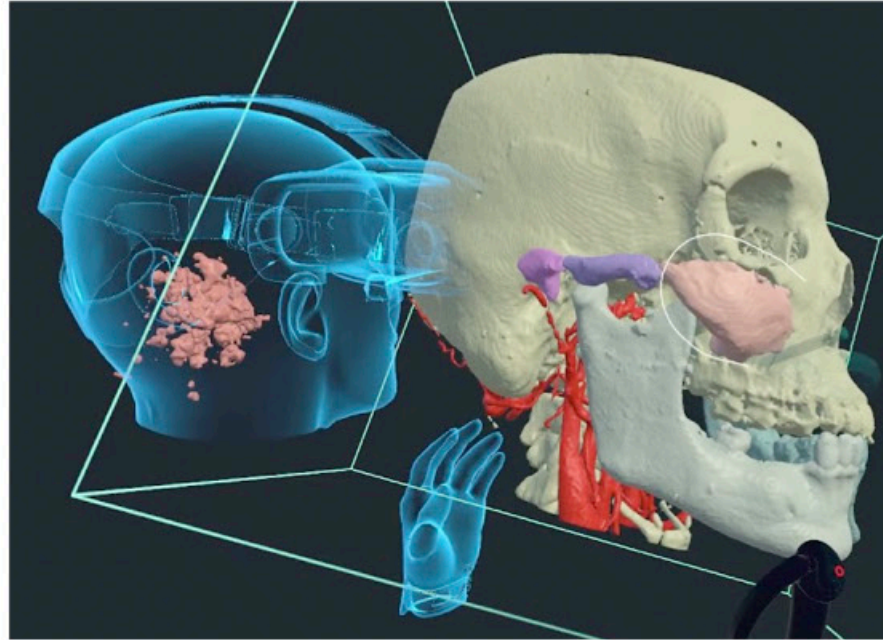
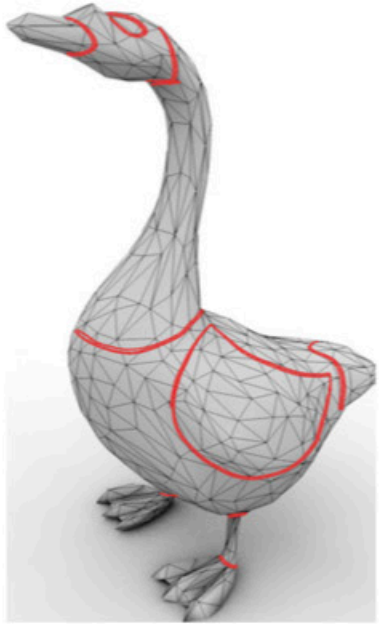
Our motivation



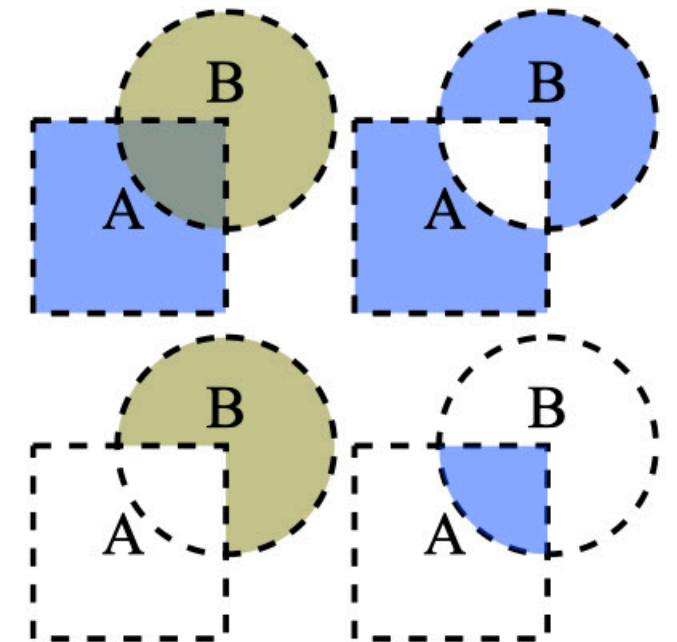
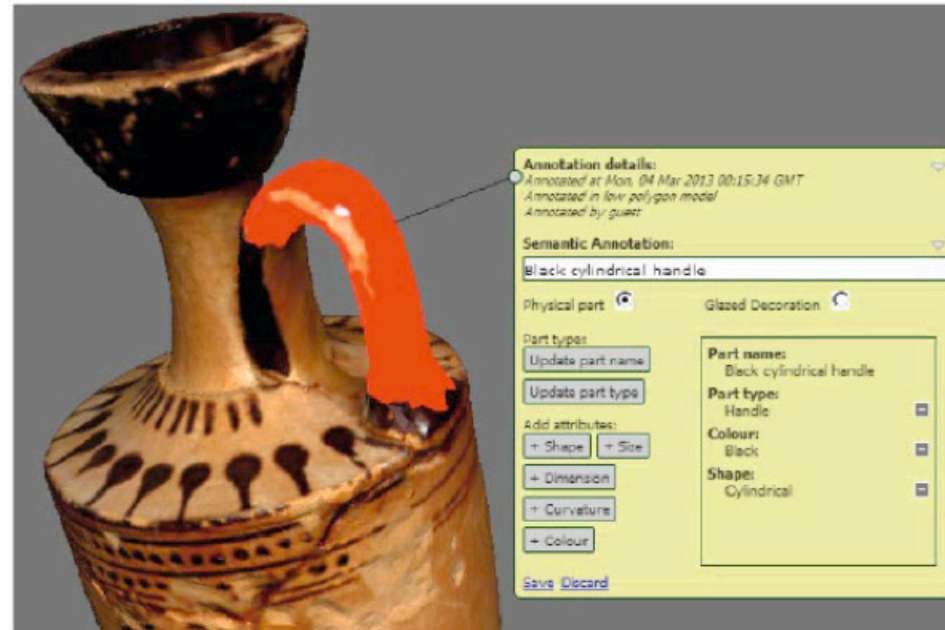
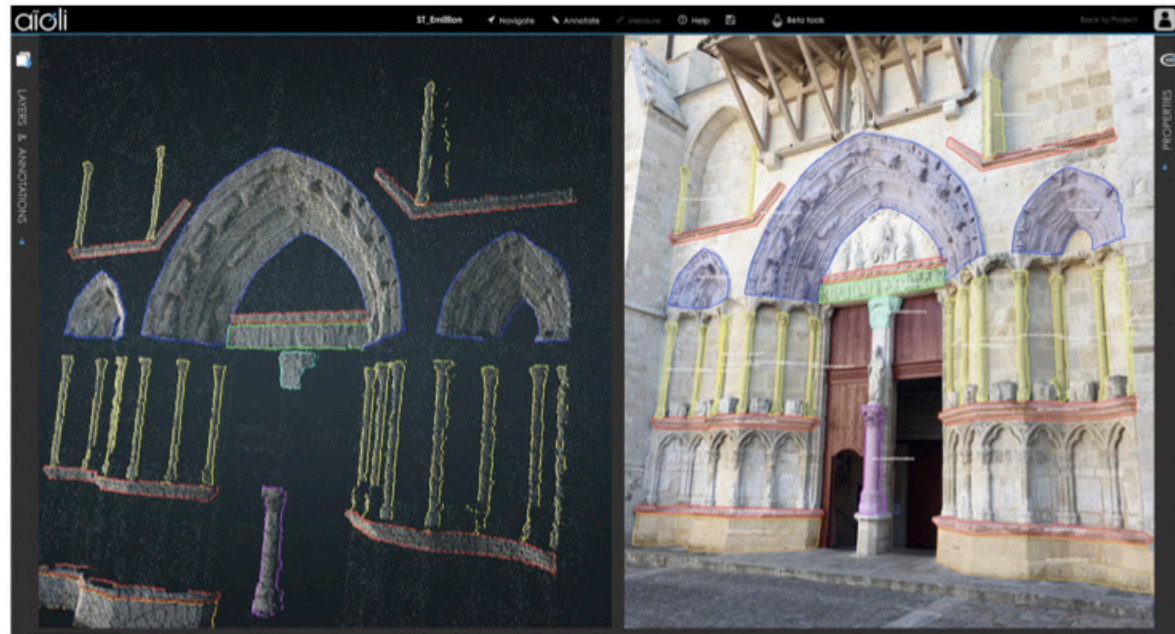
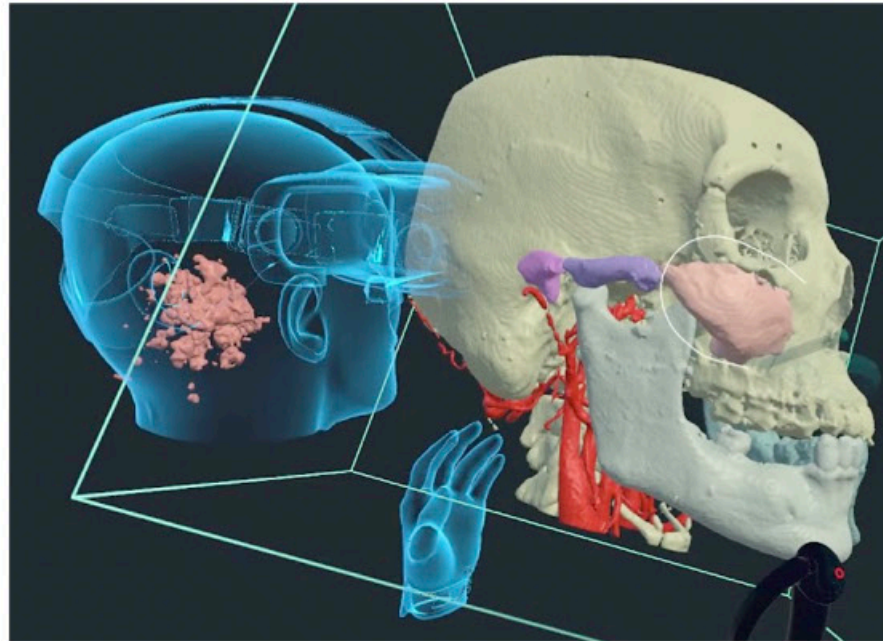
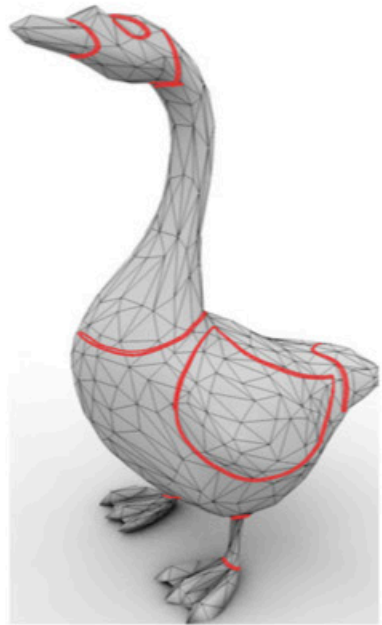
Our motivation



Our motivation



Our motivation



Past definitions: Euclidean only!

Past definitions: Euclidean only!

(Generalized) winding number = solid angle

[Euler 1781; Lagrange 1798; Gauss 1838, Maxwell 1881...]

On Solid Angles.

417.] We have already proved that at any point P the potential due to a magnetic shell is equal to the solid angle subtended by the edge of the shell multiplied by the strength

[Maxwell 1881]

Past definitions: Euclidean only!

(Generalized) winding number = solid angle

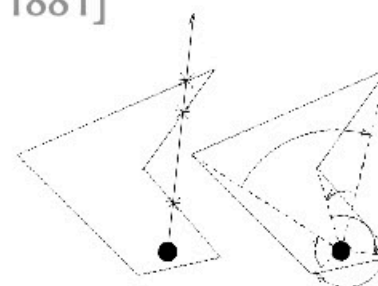
[Euler 1781; Lagrange 1798; Gauss 1838, Maxwell 1881...]

Winding number & solid angle in graphics

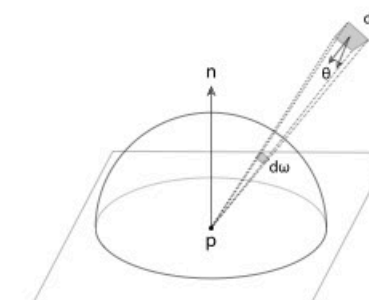
[Shimrat 1962; Haines 1994; Goral et al. 1984; Veach & Guibas 1995...]

On Solid Angles.
417.] We have already proved that at any point P the potential due to a magnetic shell is equal to the solid angle subtended by the edge of the shell multiplied by the strength

[Maxwell 1881]



[Haines 1994]



[Pharr et al. 2018]

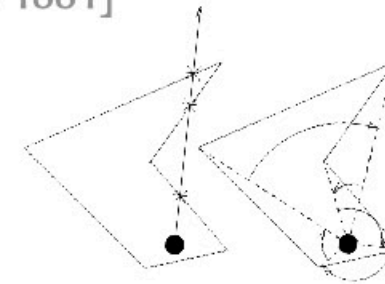
Past definitions: Euclidean only!

(Generalized) winding number = solid angle

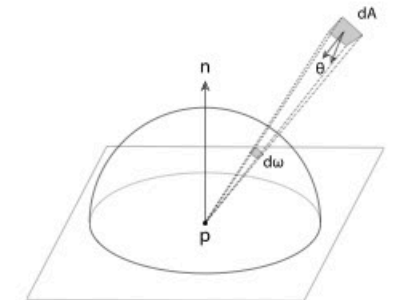
[Euler 1781; Lagrange 1798; Gauss 1838, Maxwell 1881...]

On Solid Angles.
417.] We have already proved that at any point P the potential due to a magnetic shell is equal to the solid angle subtended by the edge of the shell multiplied by the strength

[Maxwell 1881]



[Haines 1994]



[Pharr et al. 2018]

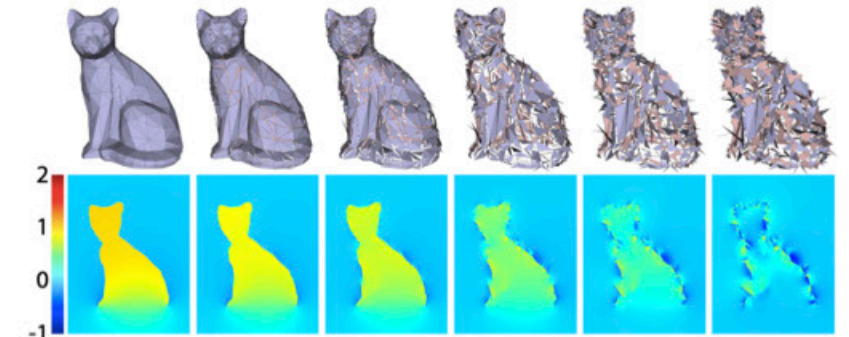
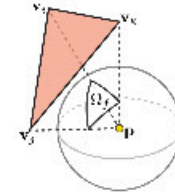
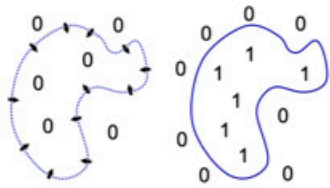
Winding number & solid angle in graphics

[Shimrat 1962; Haines 1994; Goral et al. 1984; Veach & Guibas 1995...]

Poisson Surface Reconstruction = Generalized Winding Number

[Kazhdan et al. 2006]

[Jacobson et al. 2013]



[Jacobson et al. 2013]

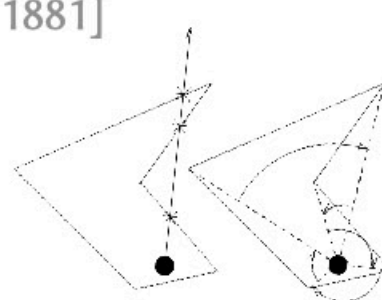
Past definitions: Euclidean only!

(Generalized) winding number = solid angle

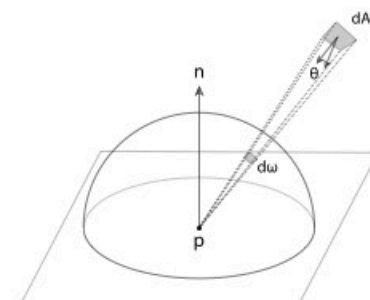
[Euler 1781; Lagrange 1798; Gauss 1838, Maxwell 1881...]

On Solid Angles.
417.] We have already proved that at any point P the potential due to a magnetic shell is equal to the solid angle subtended by the edge of the shell multiplied by the strength

[Maxwell 1881]



[Haines 1994]



[Pharr et al. 2018]

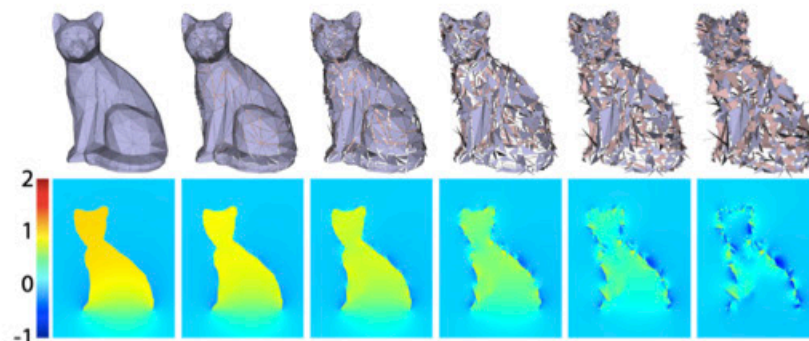
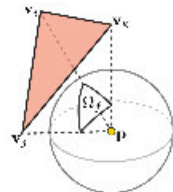
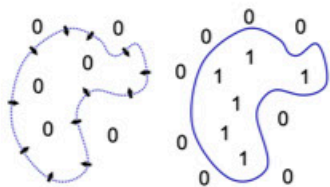
Winding number & solid angle in graphics

[Shimrat 1962; Haines 1994; Goral et al. 1984; Veach & Guibas 1995...]

Poisson Surface Reconstruction = Generalized Winding Number

[Kazhdan et al. 2006]

[Jacobson et al. 2013]



[Jacobson et al. 2013]

Winding Turning number on surfaces

[Reinhart 1960, 1963; Chillingworth 1972; Humphries & Johnson 1989;

McIntyre & Cairns 1993; Chernov & Rudyak 2009]

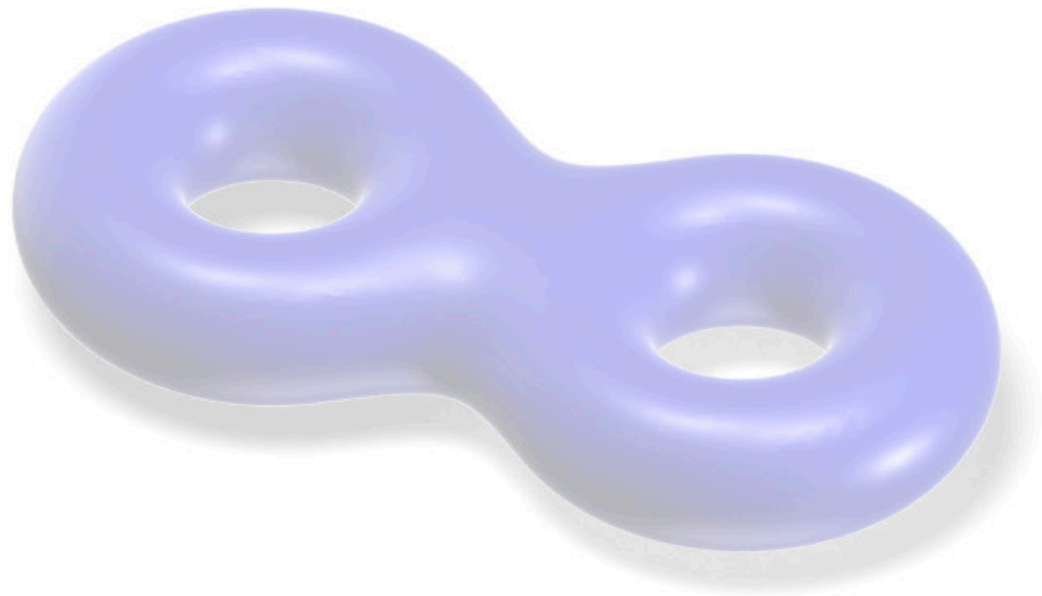
Classic methods fail

Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.

Poisson Surface Reconstruction. Kazhdan, Bolitho, Hoppe (2006)
Robust Inside-Outside Segmentation using Generalized Winding Numbers.
Jacobson, Kavan, Sorkine-Hornung (2013)

Classic methods fail

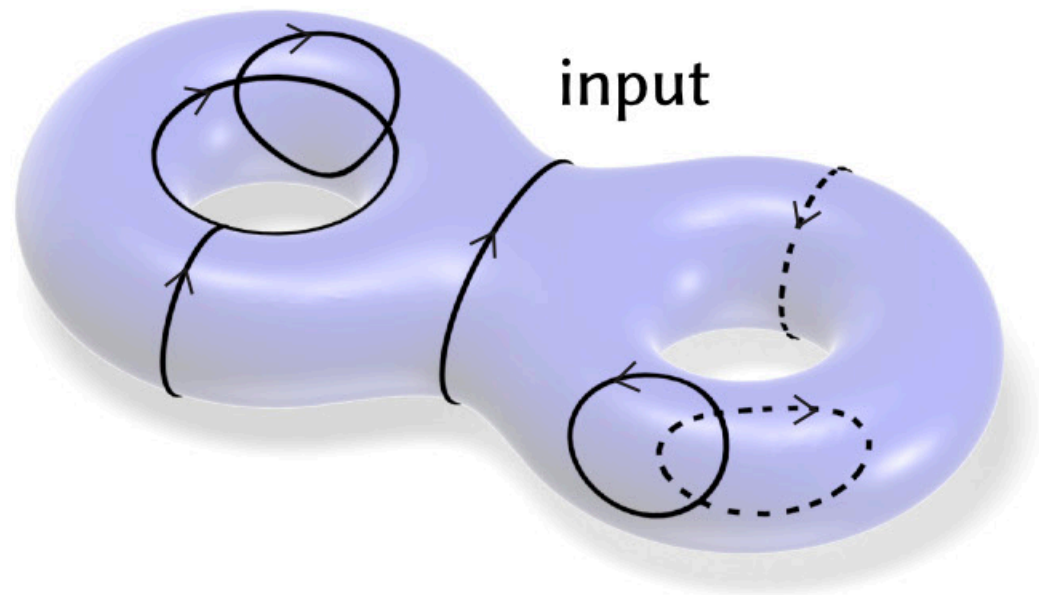
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



Poisson Surface Reconstruction. Kazhdan, Bolitho, Hoppe (2006)
Robust Inside-Outside Segmentation using Generalized Winding Numbers.
Jacobson, Kavan, Sorkine-Hornung (2013)

Classic methods fail

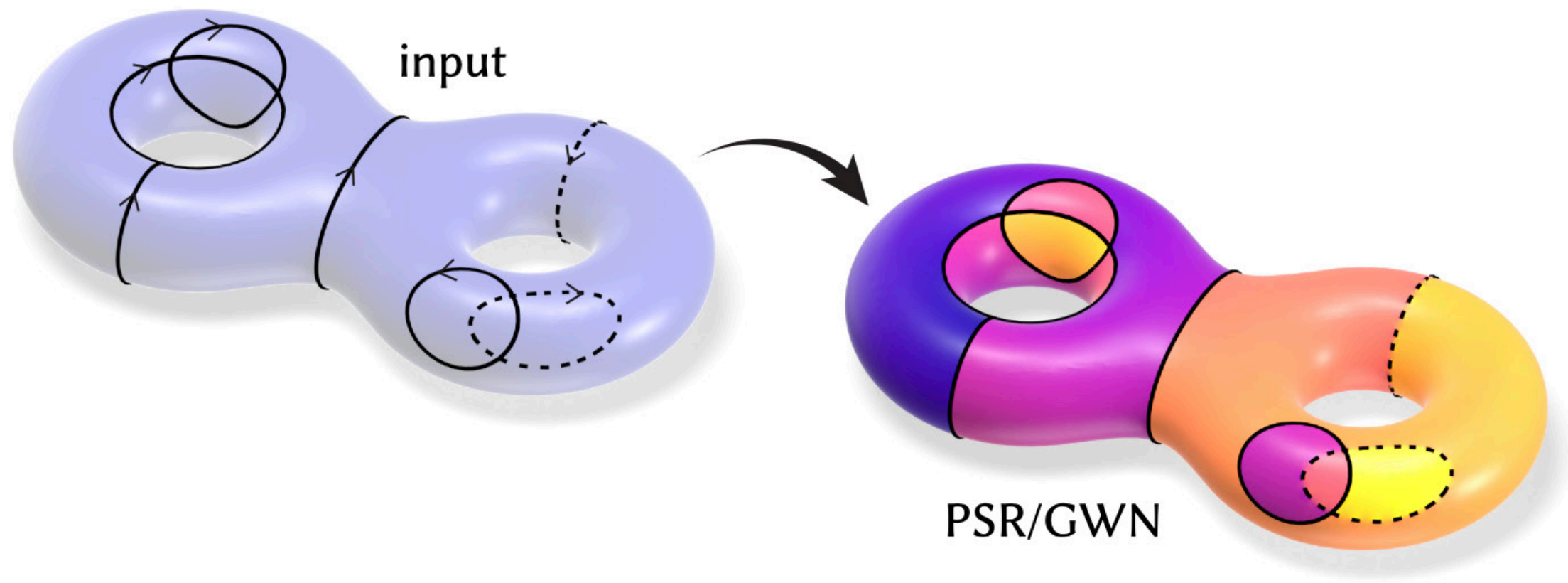
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



Poisson Surface Reconstruction. Kazhdan, Bolitho, Hoppe (2006)
Robust Inside-Outside Segmentation using Generalized Winding Numbers.
Jacobson, Kavan, Sorkine-Hornung (2013)

Classic methods fail

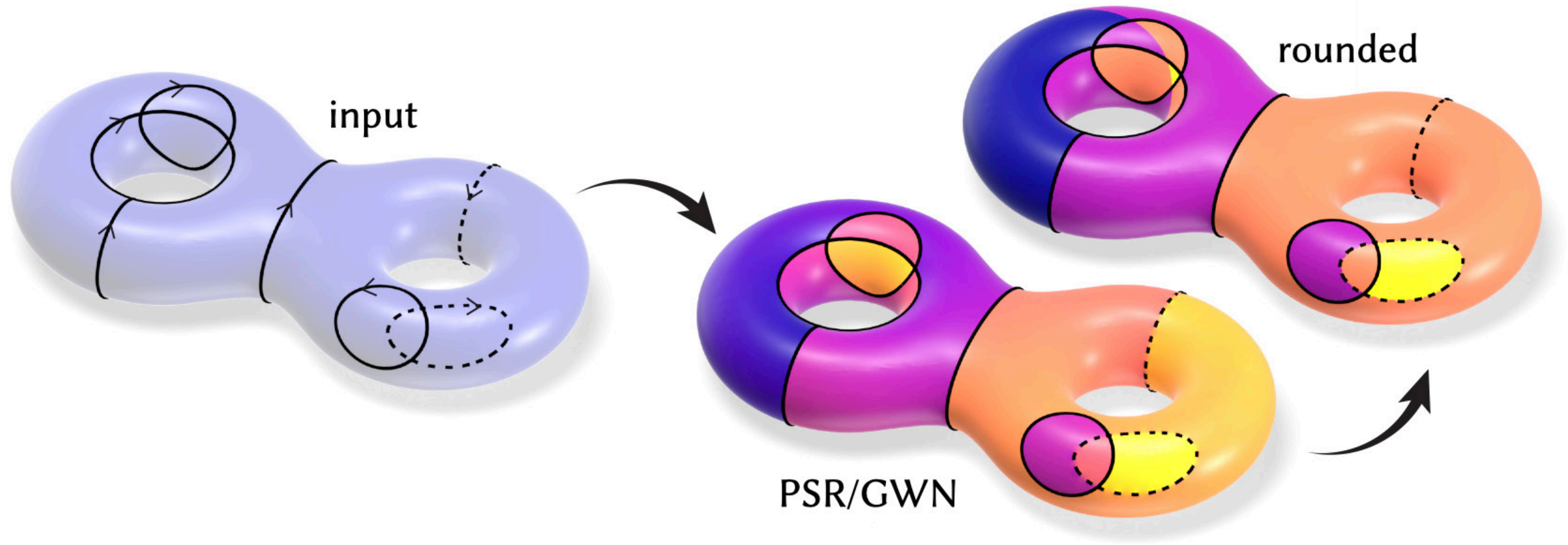
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



Poisson Surface Reconstruction. Kazhdan, Bolitho, Hoppe (2006)
Robust Inside-Outside Segmentation using Generalized Winding Numbers.
Jacobson, Kavan, Sorkine-Hornung (2013)

Classic methods fail

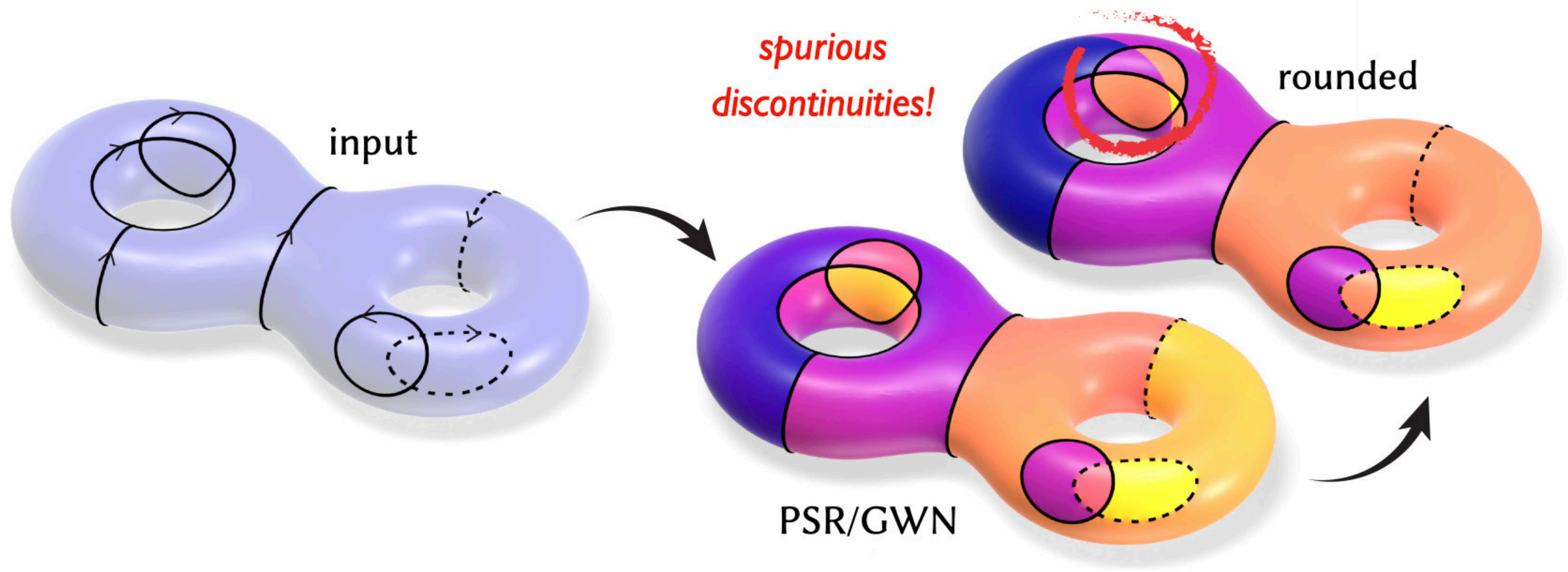
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



Poisson Surface Reconstruction. Kazhdan, Bolitho, Hoppe (2006)
Robust Inside-Outside Segmentation using Generalized Winding Numbers.
Jacobson, Kavan, Sorkine-Hornung (2013)

Classic methods fail

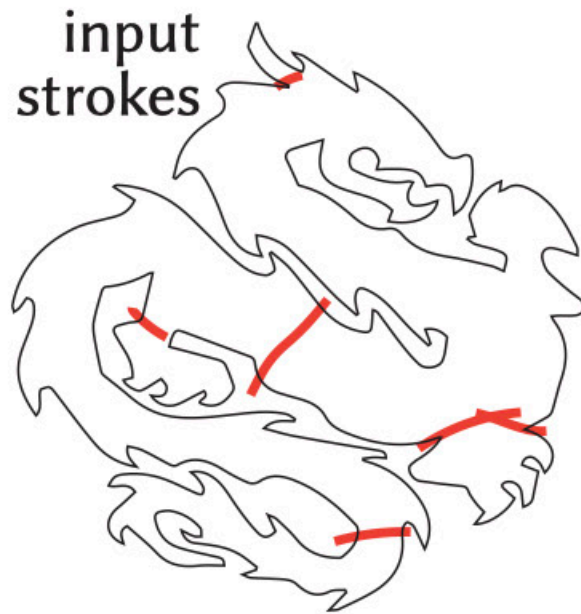
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



Poisson Surface Reconstruction. Kazhdan, Bolitho, Hoppe (2006)
Robust Inside-Outside Segmentation using Generalized Winding Numbers.
Jacobson, Kavan, Sorkine-Hornung (2013)

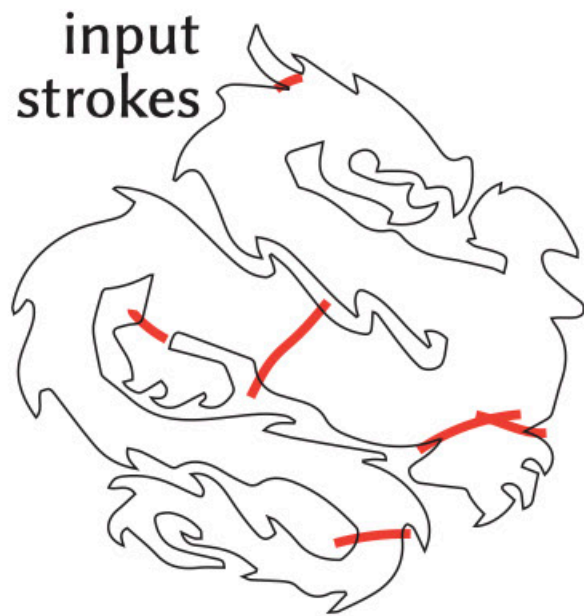
Classic methods fail

Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



Classic methods fail

Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.

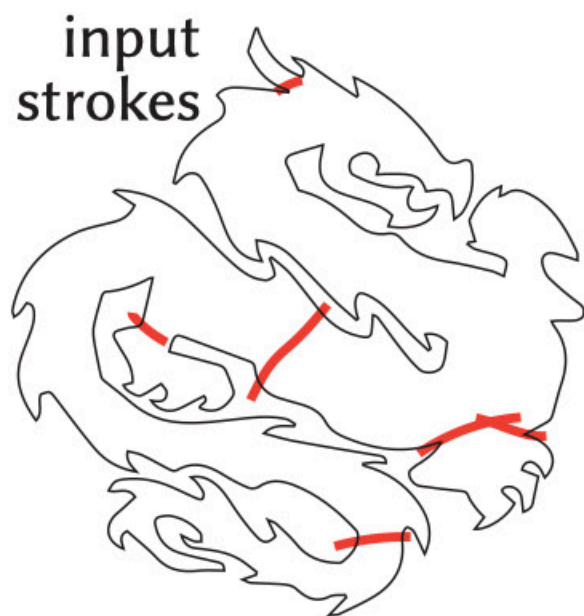


GWN



Classic methods fail

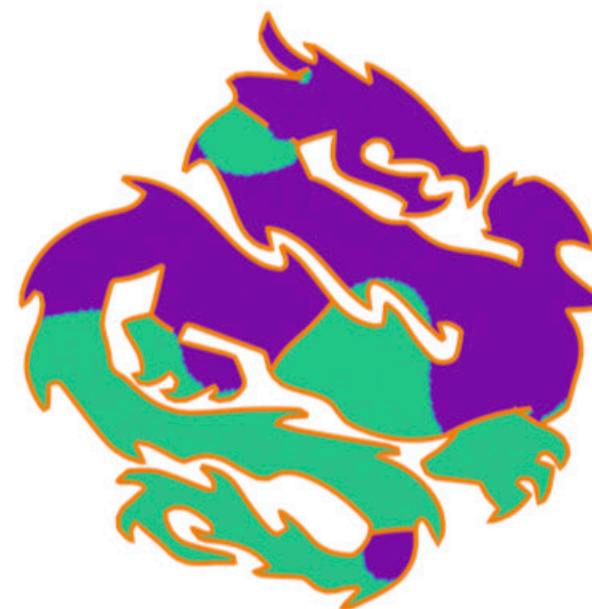
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



GWN

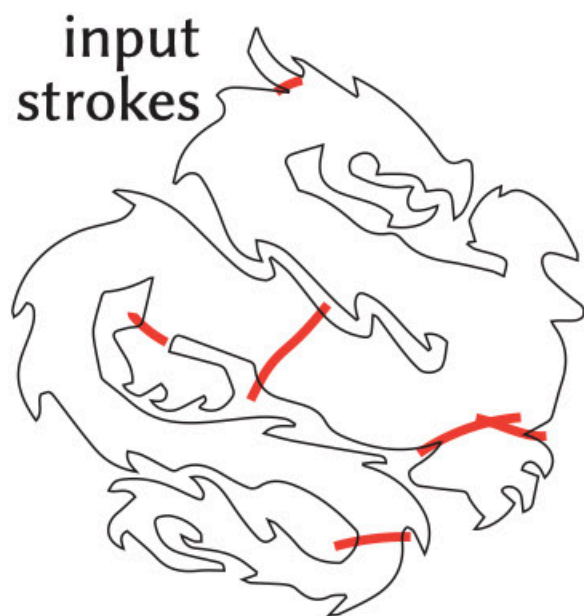


rounded function



Classic methods fail

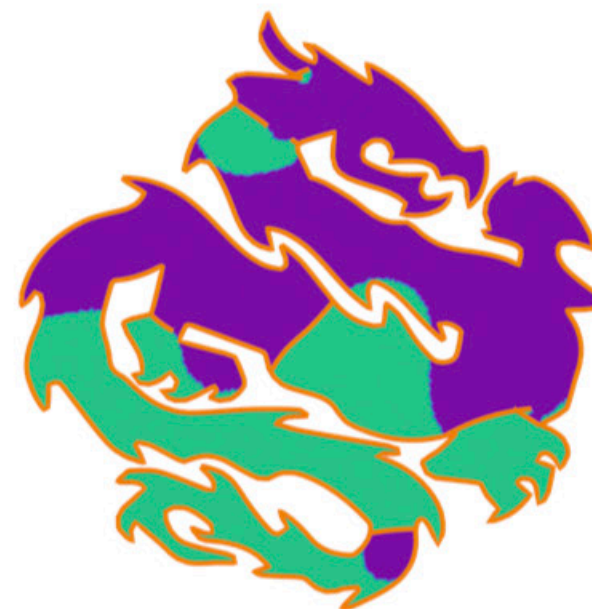
Poisson Surface Reconstruction (PSR) & Generalized Winding Number (GWN)
don't always identify well-defined regions.



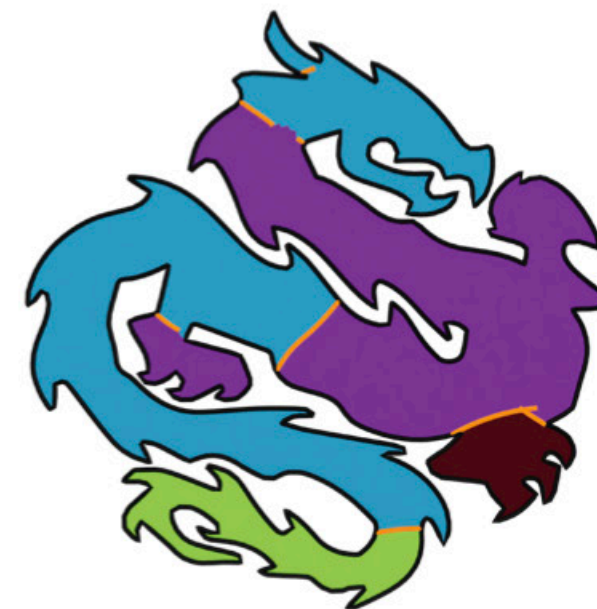
GWN



rounded function

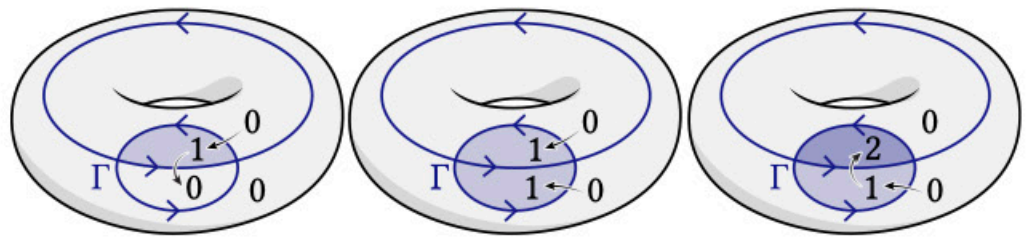
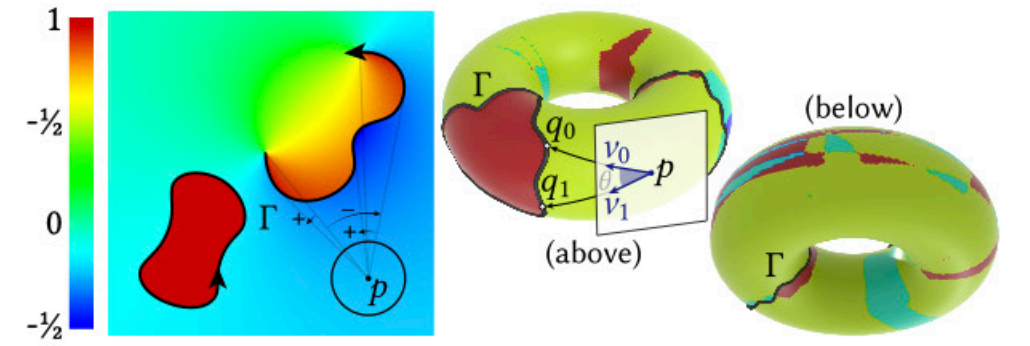
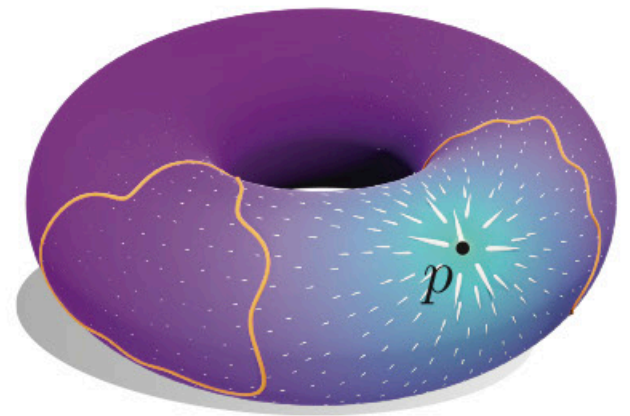
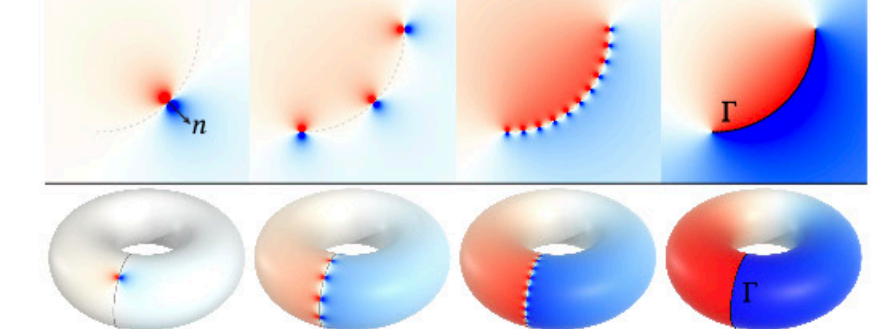
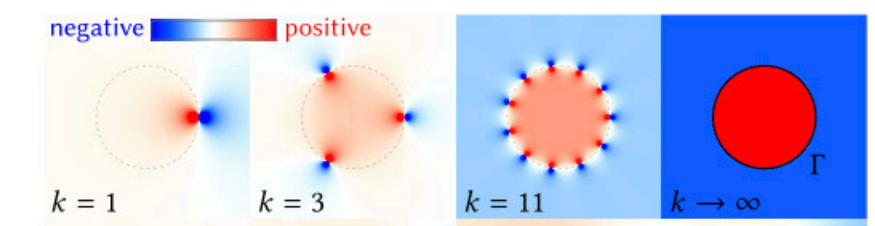
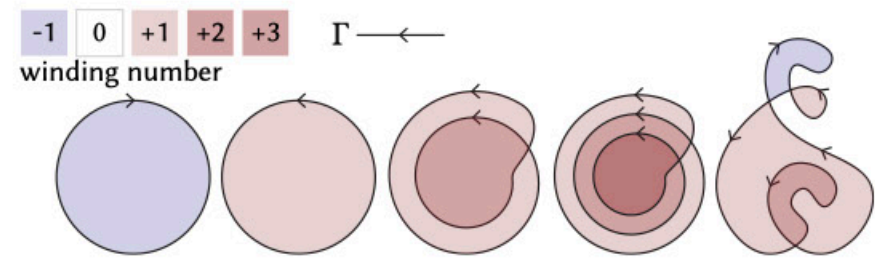
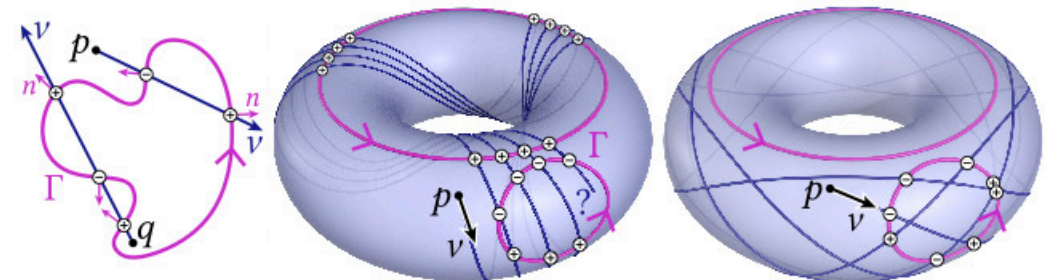
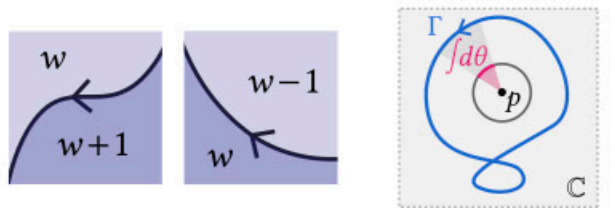
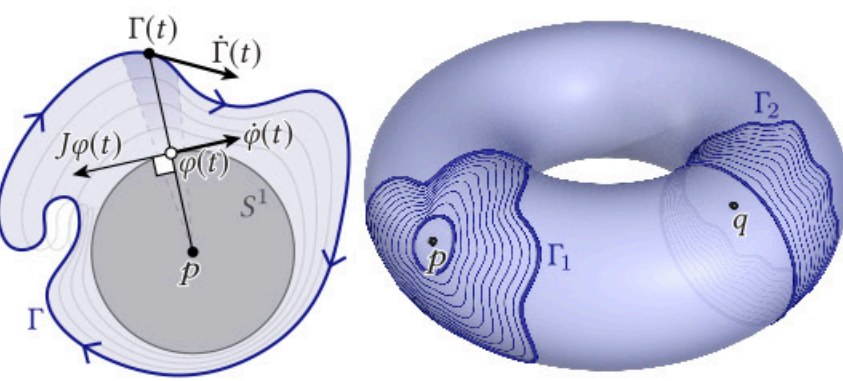
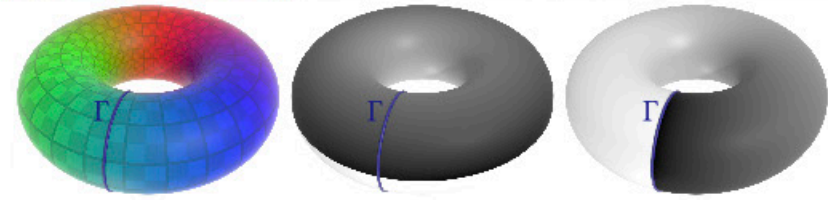
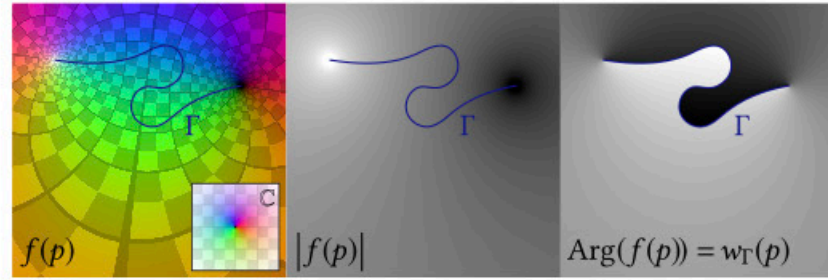


ours



Many formulas for solid angle...

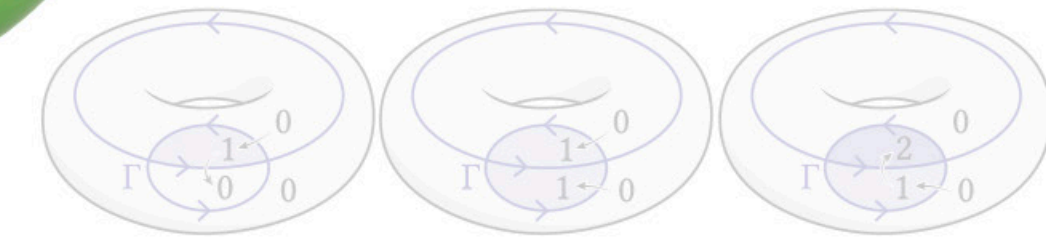
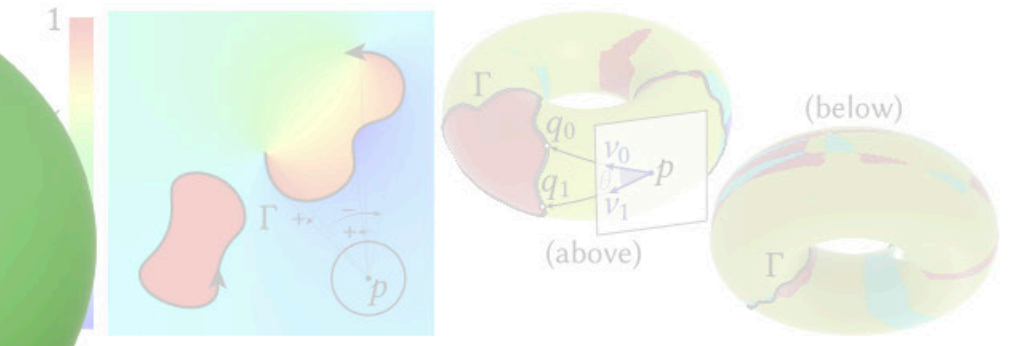
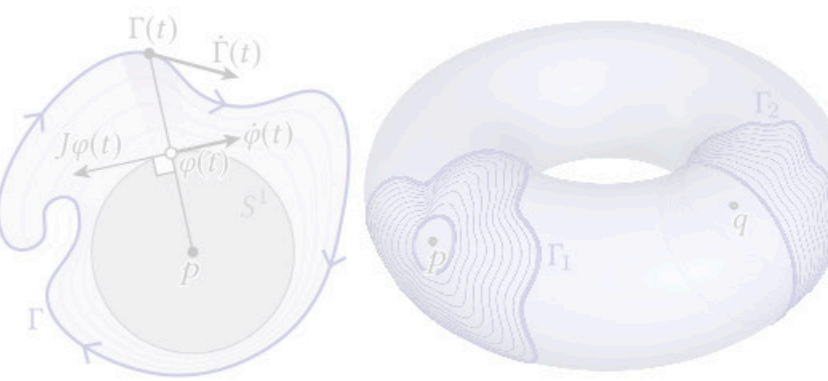
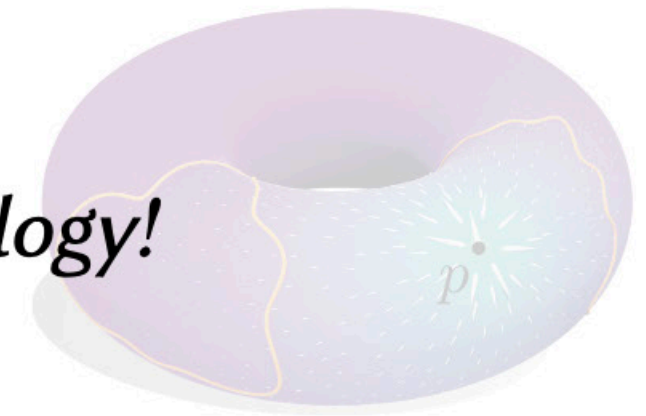
Many formulas for solid angle...



Many formulas for solid angle...



... none work on domains of general topology!



The general case is difficult!

The general case is difficult!

BoolSurf [Riso et al. 2022]:



[Riso et al. 2022]

The general case is difficult!

BoolSurf [Riso et al. 2022]:



input is already segmented into loops

[Riso et al. 2022]

The general case is difficult!

BoolSurf [Riso et al. 2022]:



[Riso et al. 2022]

input is already segmented into loops

closed loops only

The general case is difficult!

BoolSurf [Riso et al. 2022]:

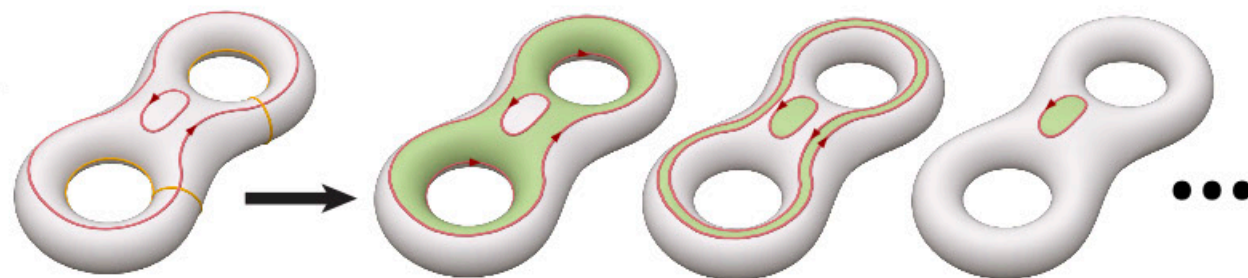


[Riso et al. 2022]

input is already segmented into loops

closed loops only

requires user guidance to handle nonbounding curves



The general case is difficult!

BoolSurf [Riso et al. 2022]:

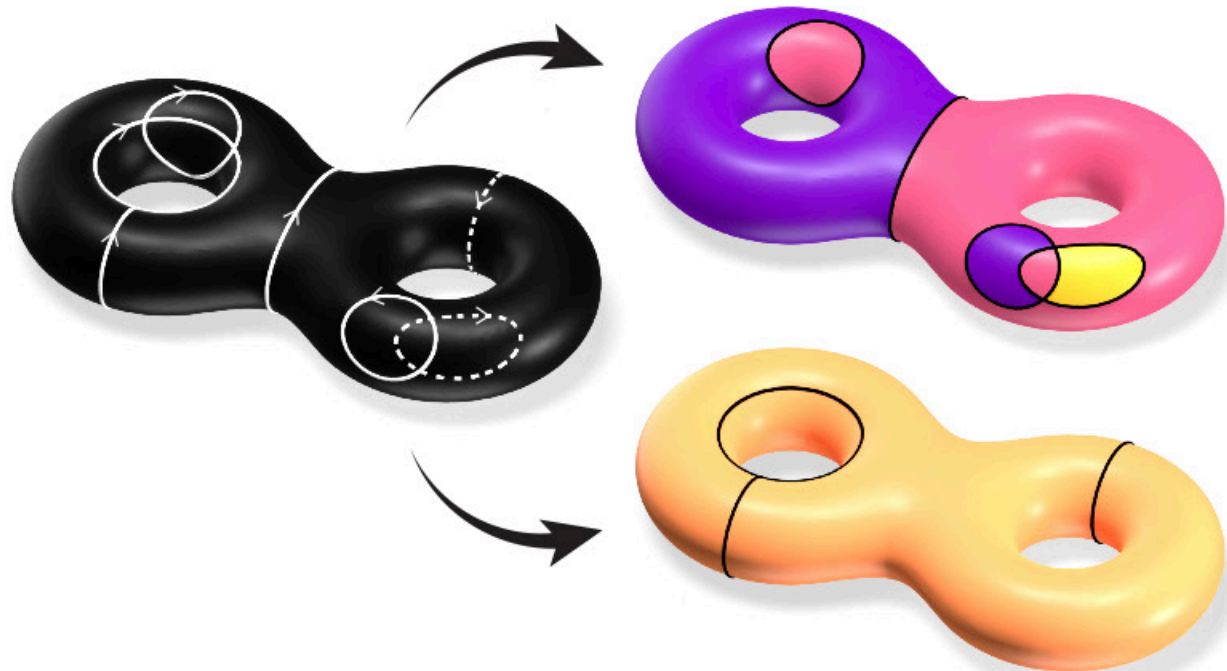
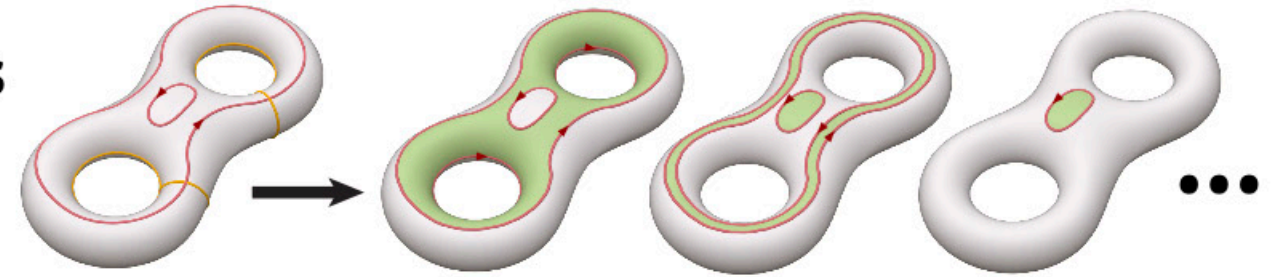


[Riso et al. 2022]

input is already segmented into loops

closed loops only

requires user guidance to handle nonbounding curves



The general case is difficult!

BoolSurf [Riso et al. 2022]:

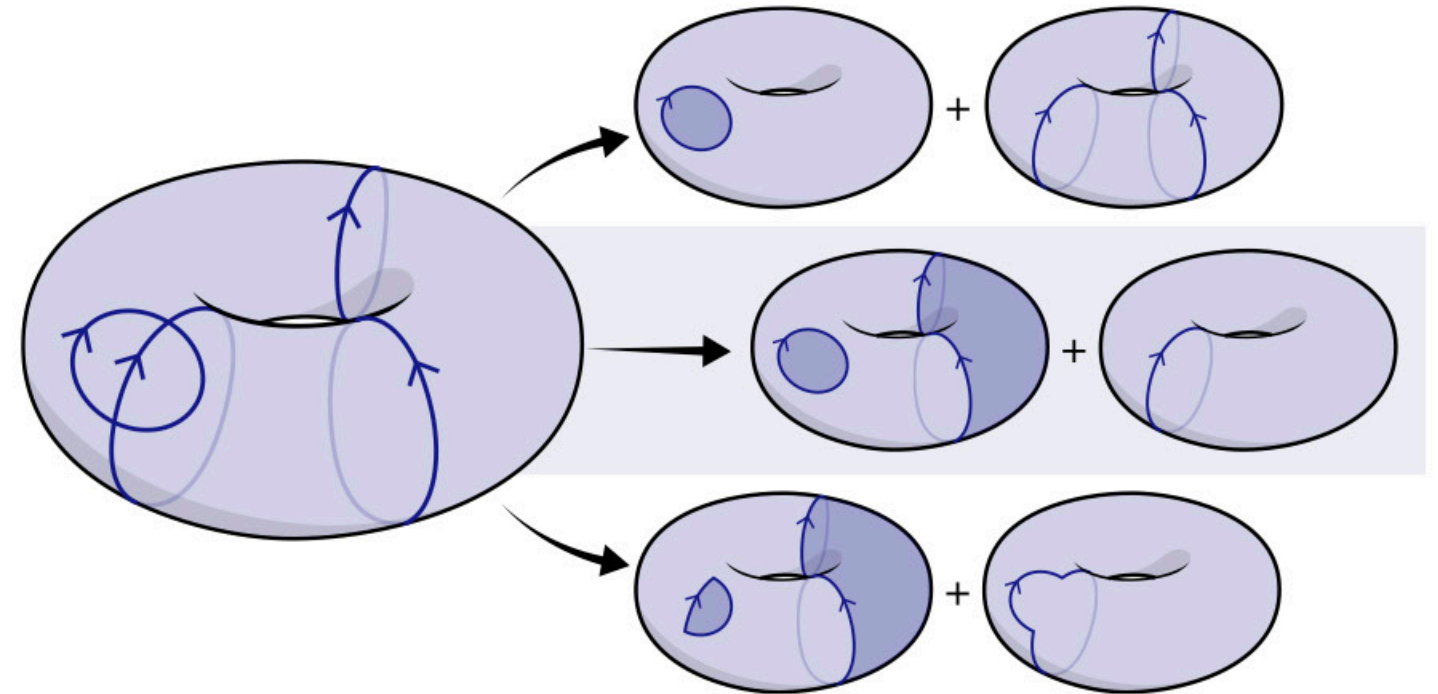
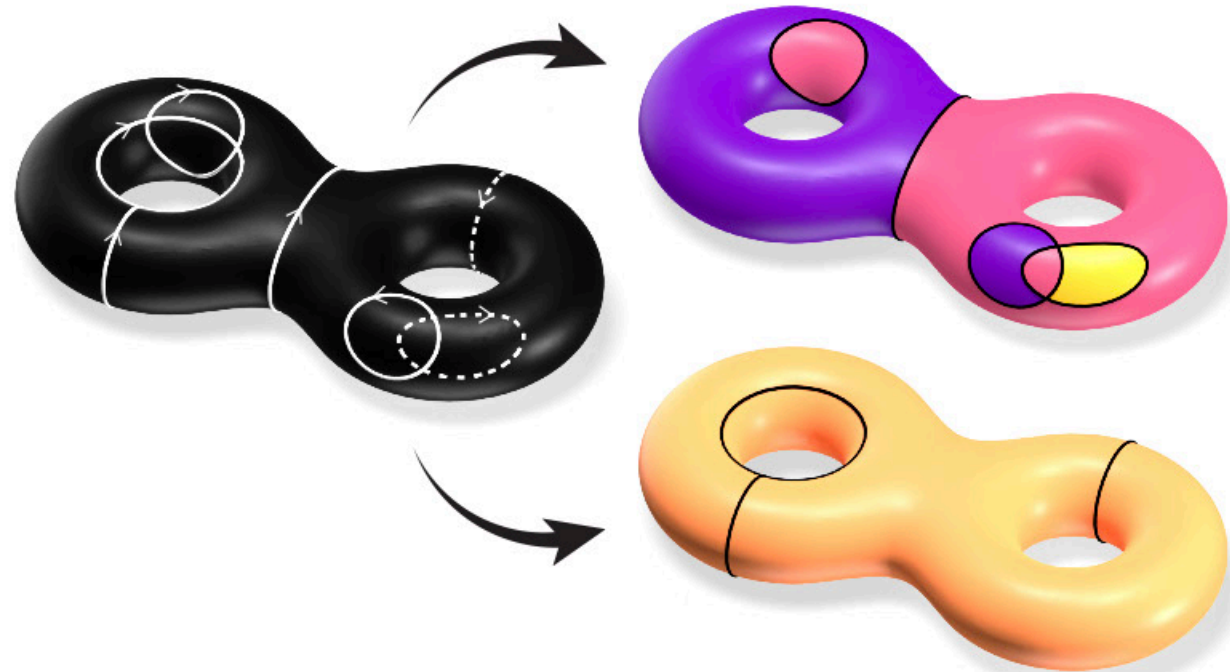
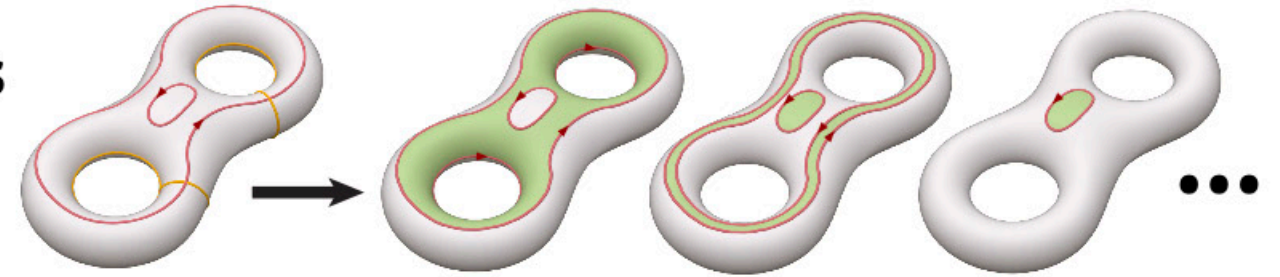


[Riso et al. 2022]

input is already segmented into loops

closed loops only

requires user guidance to handle nonbounding curves



The general case is difficult!

BoolSurf [Riso et al. 2022]:

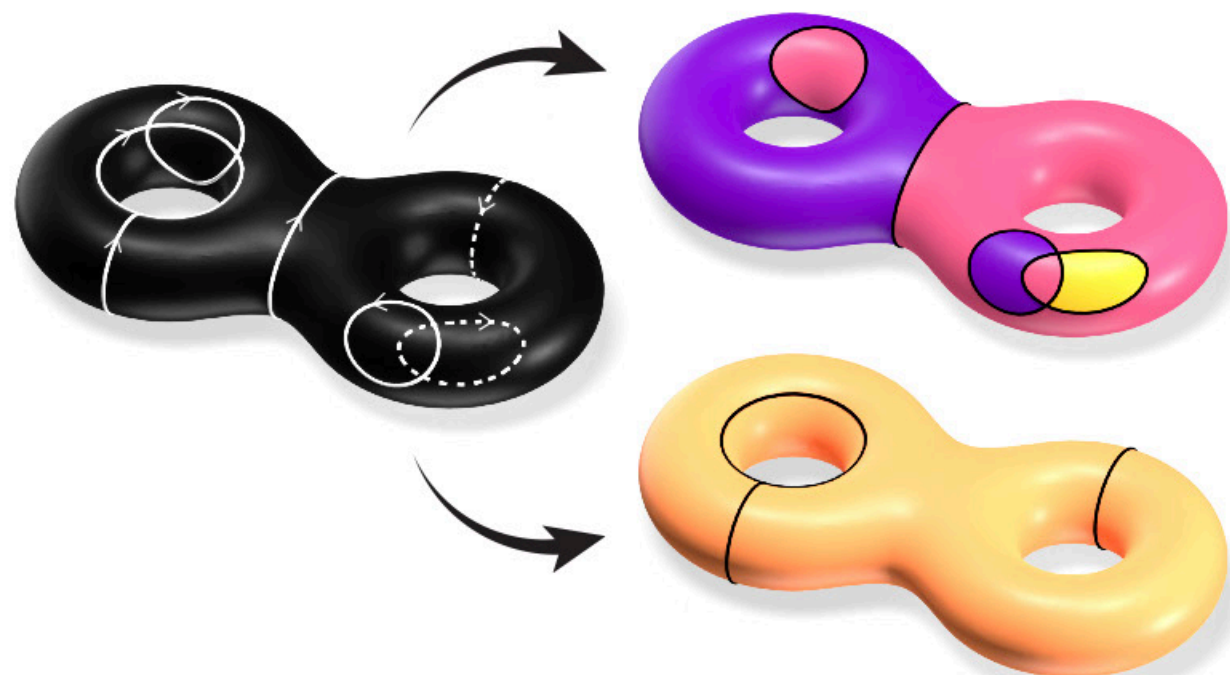
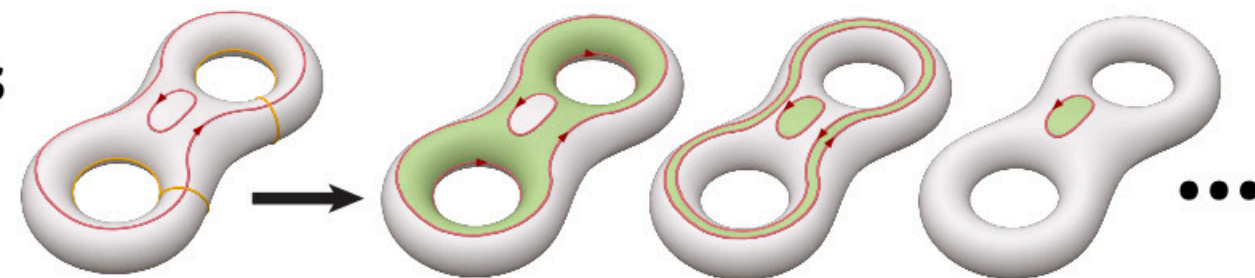


[Riso et al. 2022]

input is already segmented into loops

closed loops only

requires user guidance to handle nonbounding curves



Past approaches to "homological geometry processing"

Past approaches to "homological geometry processing"

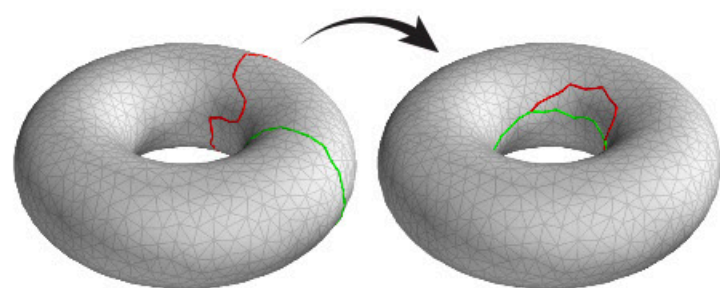
Geometry processing with homological constraints

[Born et al. 2021; Dey et al. 2010; Wang & Chern 2021]

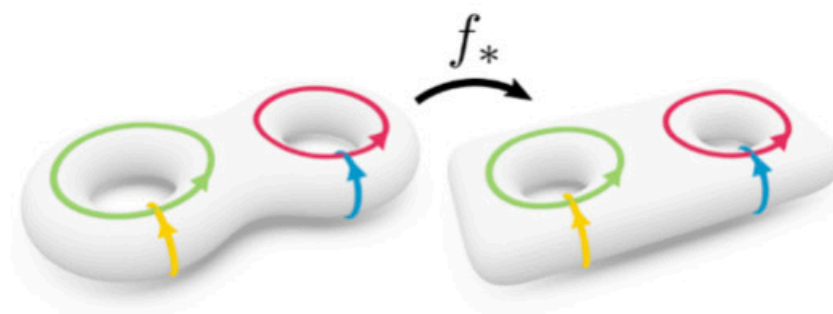
Past approaches to "homological geometry processing"

Geometry processing with homological constraints

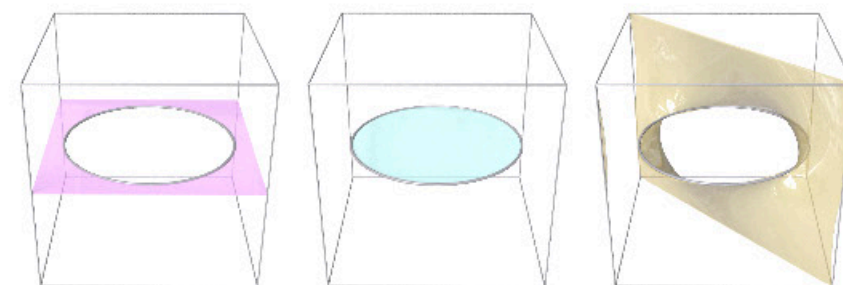
[Born et al. 2021; Dey et al. 2010; Wang & Chern 2021]



[Dey et al. 2010]



[Born et al. 2021]

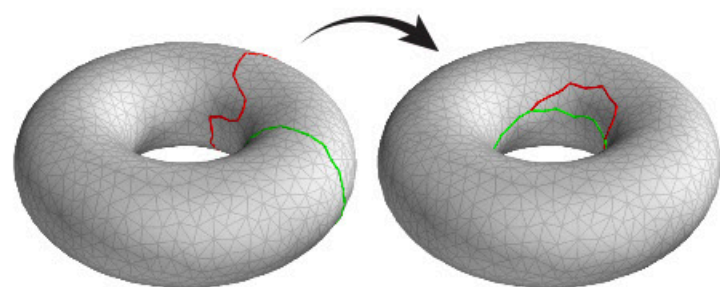


[Wang & Chern 2021]

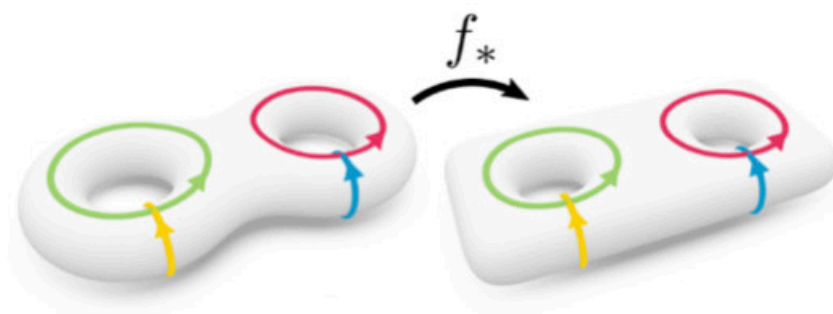
Past approaches to "homological geometry processing"

Geometry processing with homological constraints

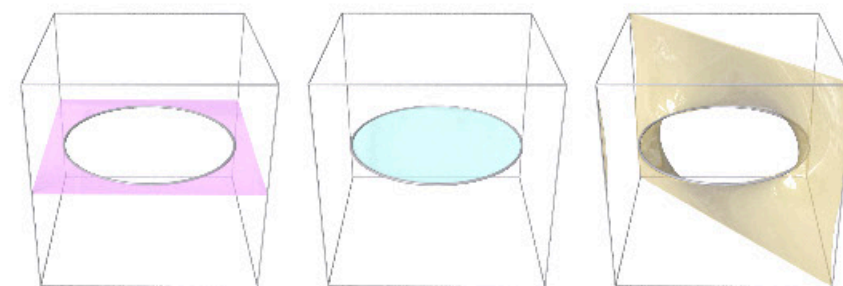
[Born et al. 2021; Dey et al. 2010; Wang & Chern 2021]



[Dey et al. 2010]

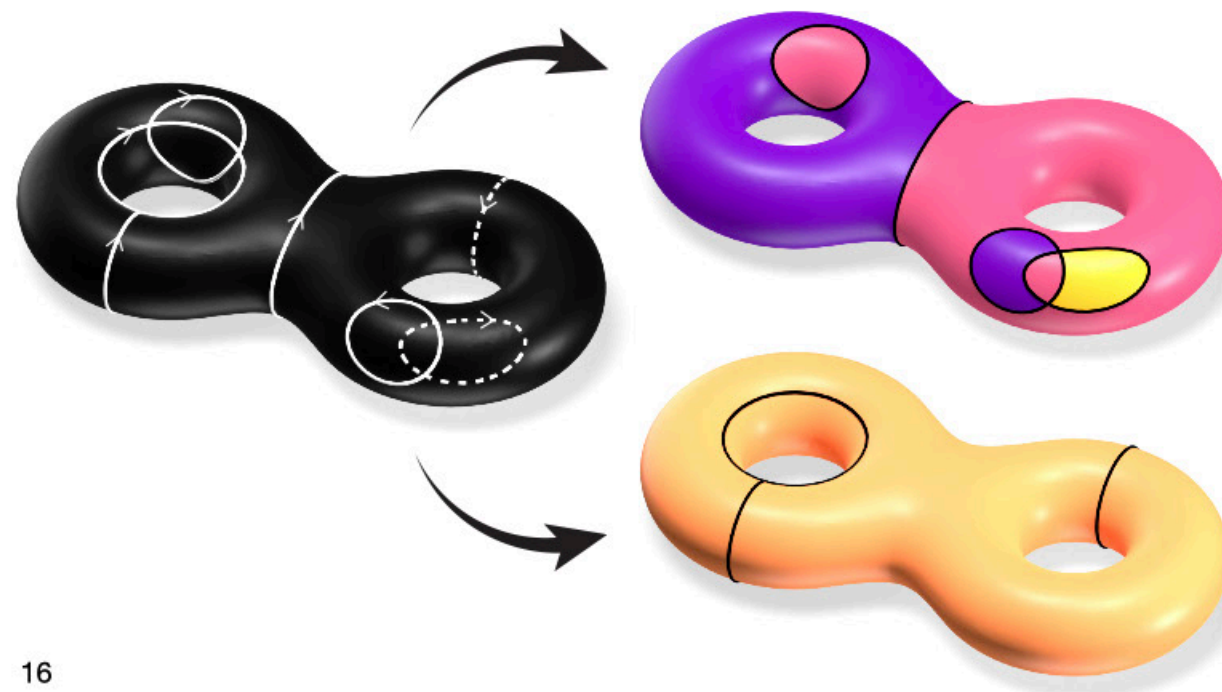


[Born et al. 2021]



[Wang & Chern 2021]

We want to *infer* curve topology!

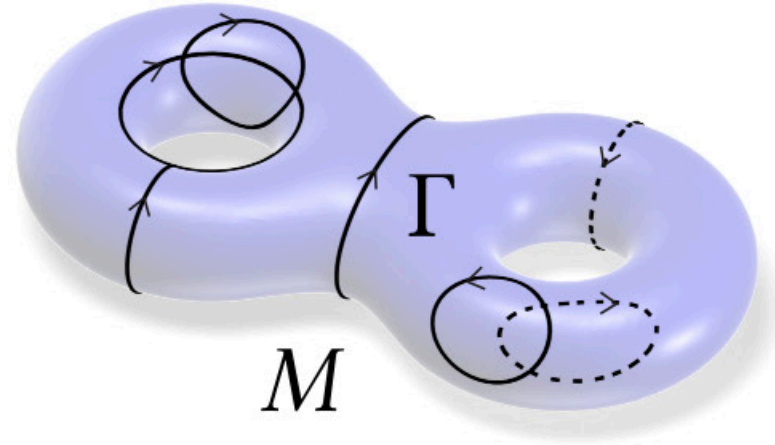


Algorithm input & output

Algorithm input & output

Input:

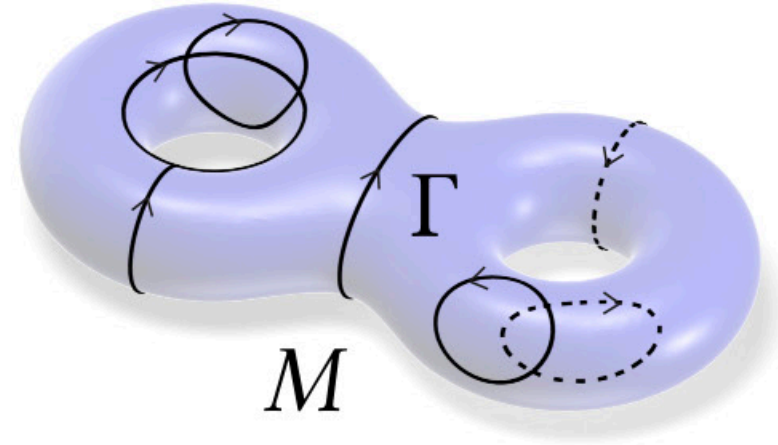
(Possibly broken) oriented curve Γ
on a surface M .



Algorithm input & output

Input:

(Possibly broken) oriented curve Γ
on a surface M .

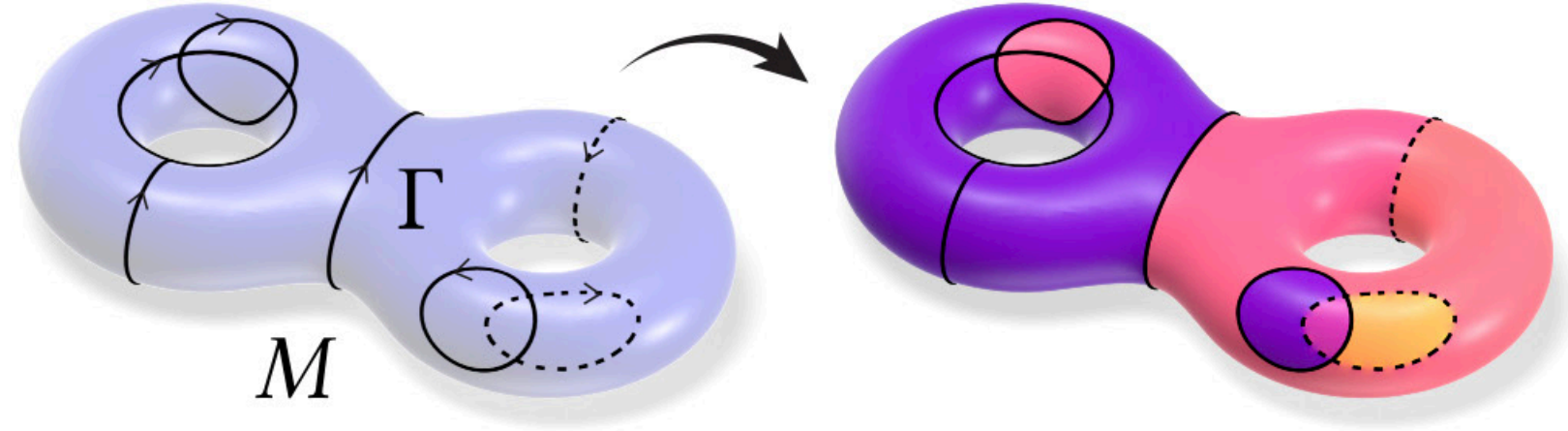


Output:

Algorithm input & output

Input:

(Possibly broken) oriented curve Γ
on a surface M .



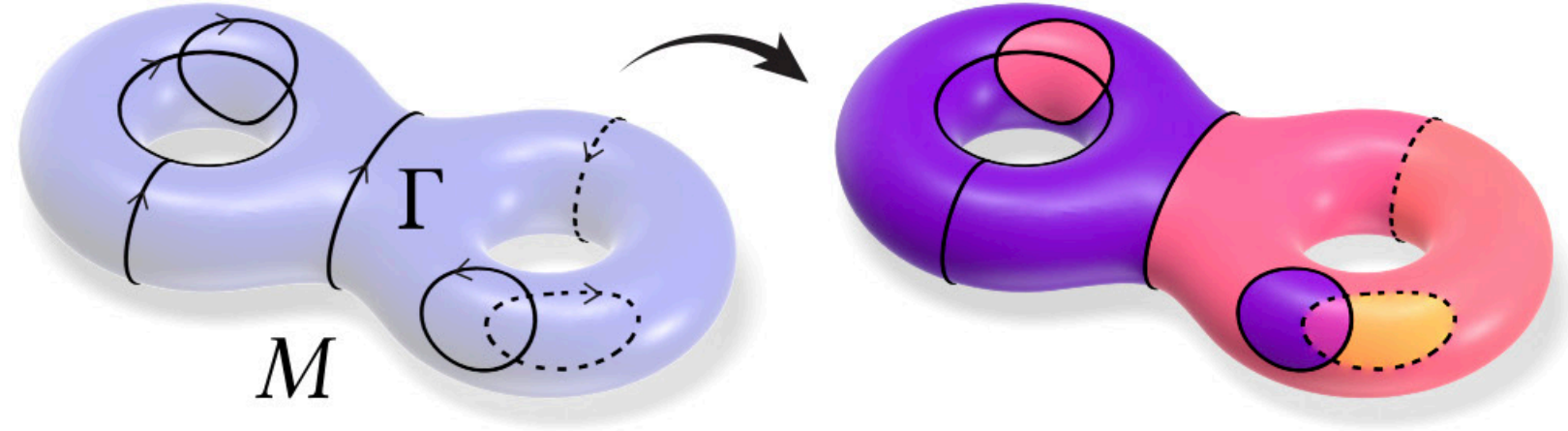
Output:

Region labels induced by bounding components of Γ .

Algorithm input & output

Input:

(Possibly broken) oriented curve Γ
on a surface M .



Output:

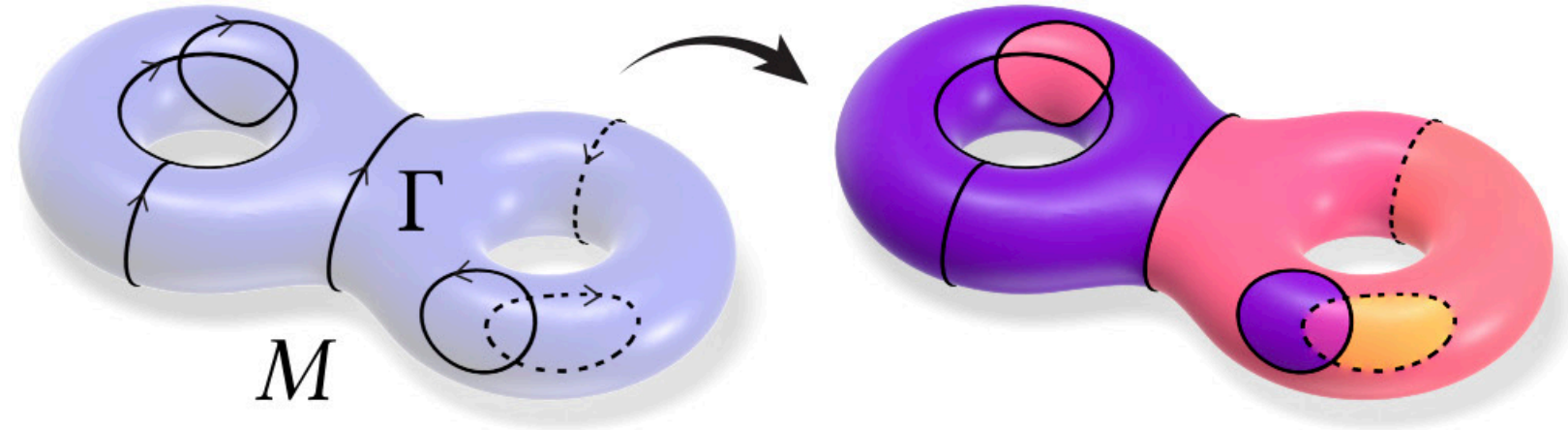
Region labels induced by bounding components of Γ .

A decomposition of Γ into:

Algorithm input & output

Input:

(Possibly broken) oriented curve Γ
on a surface M .



Output:

Region labels induced by bounding components of Γ .

A decomposition of Γ into:

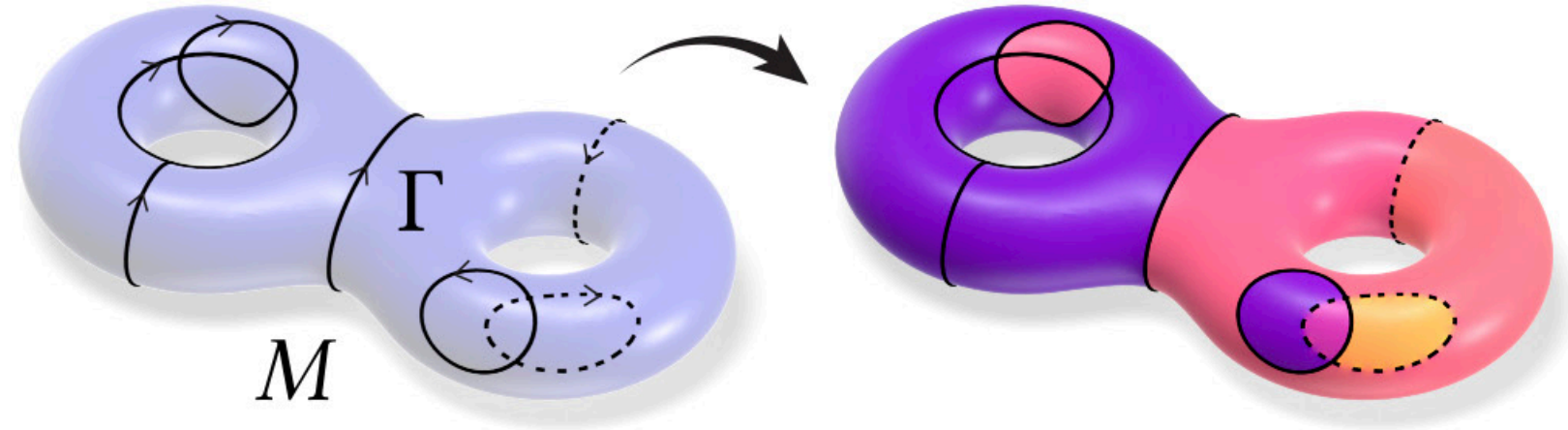
- bounding components that induce valid regions



Algorithm input & output

Input:

(Possibly broken) oriented curve Γ
on a surface M .

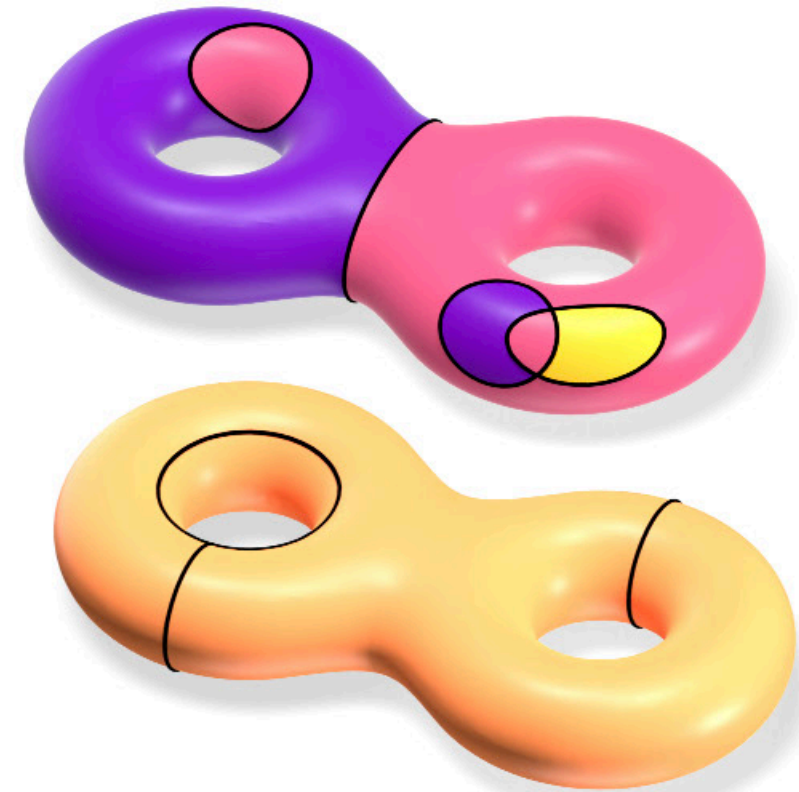


Output:

Region labels induced by bounding components of Γ .

A decomposition of Γ into:

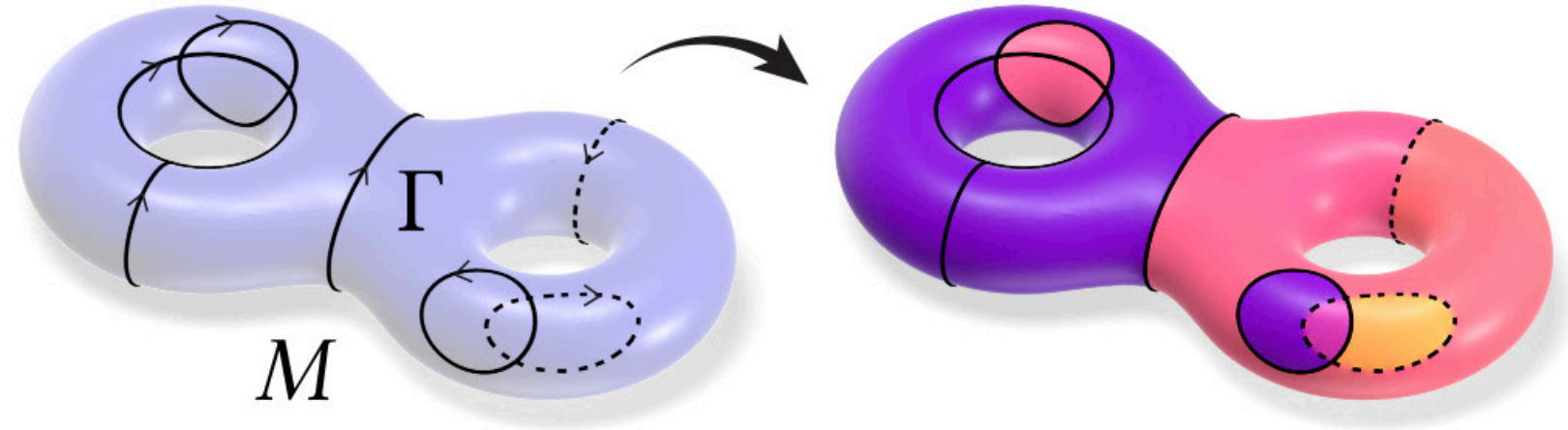
- bounding components that induce valid regions
- nonbounding components.



Algorithm input & output

Input:

(Possibly broken) oriented curve Γ on a surface M .



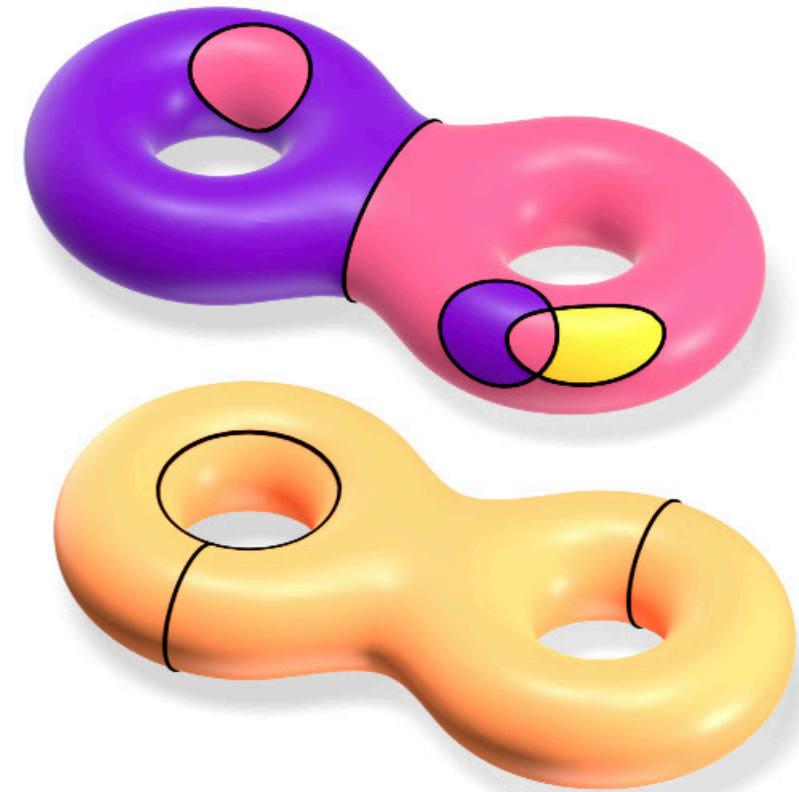
Output:

Region labels induced by bounding components of Γ .

A decomposition of Γ into:

- bounding components that induce valid regions
- nonbounding components.

A closed, completed version of the input curve.



Algorithm input & output

Input:

(Possibly broken) oriented curve Γ on a surface M .

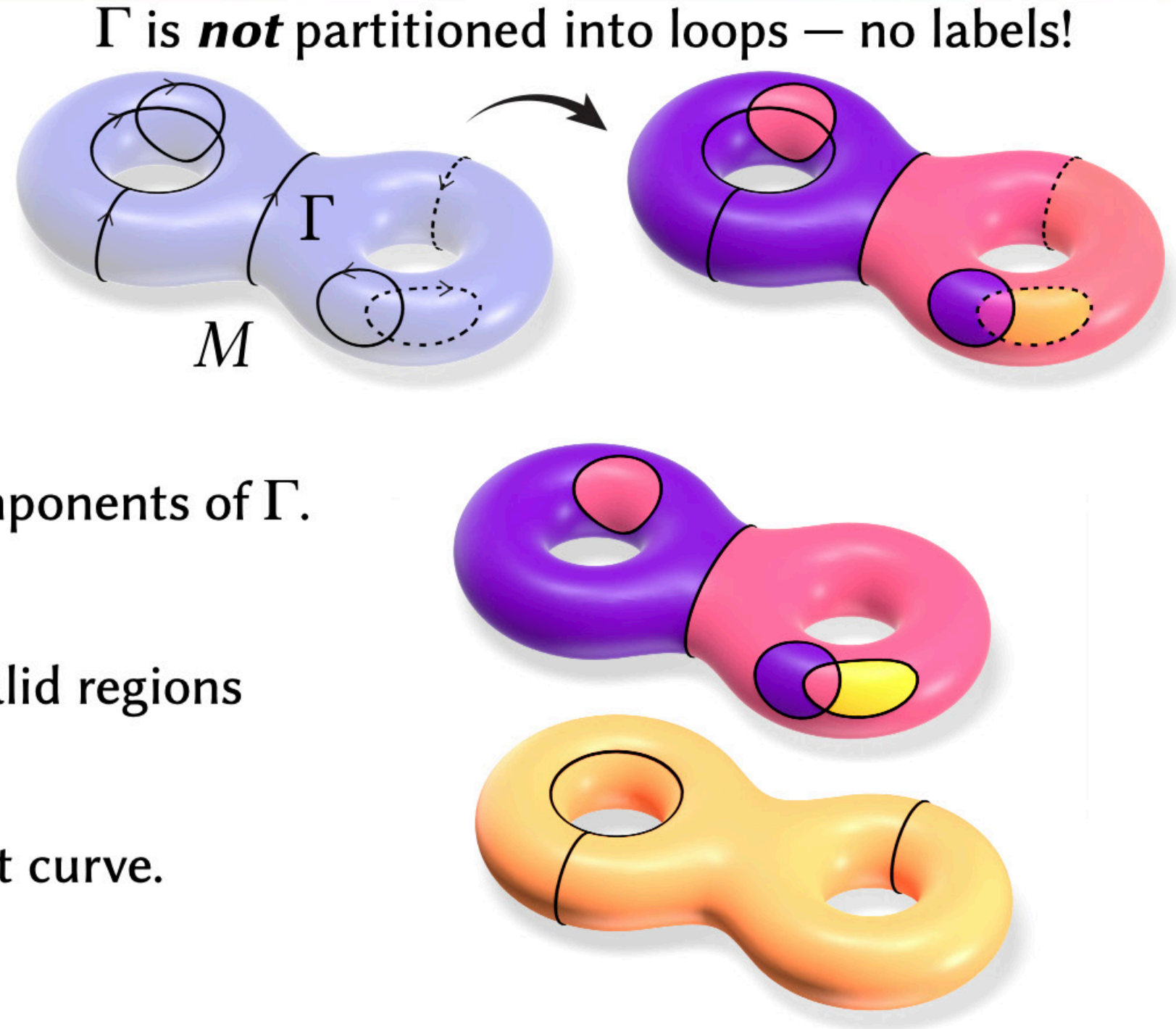
Output:

Region labels induced by bounding components of Γ .

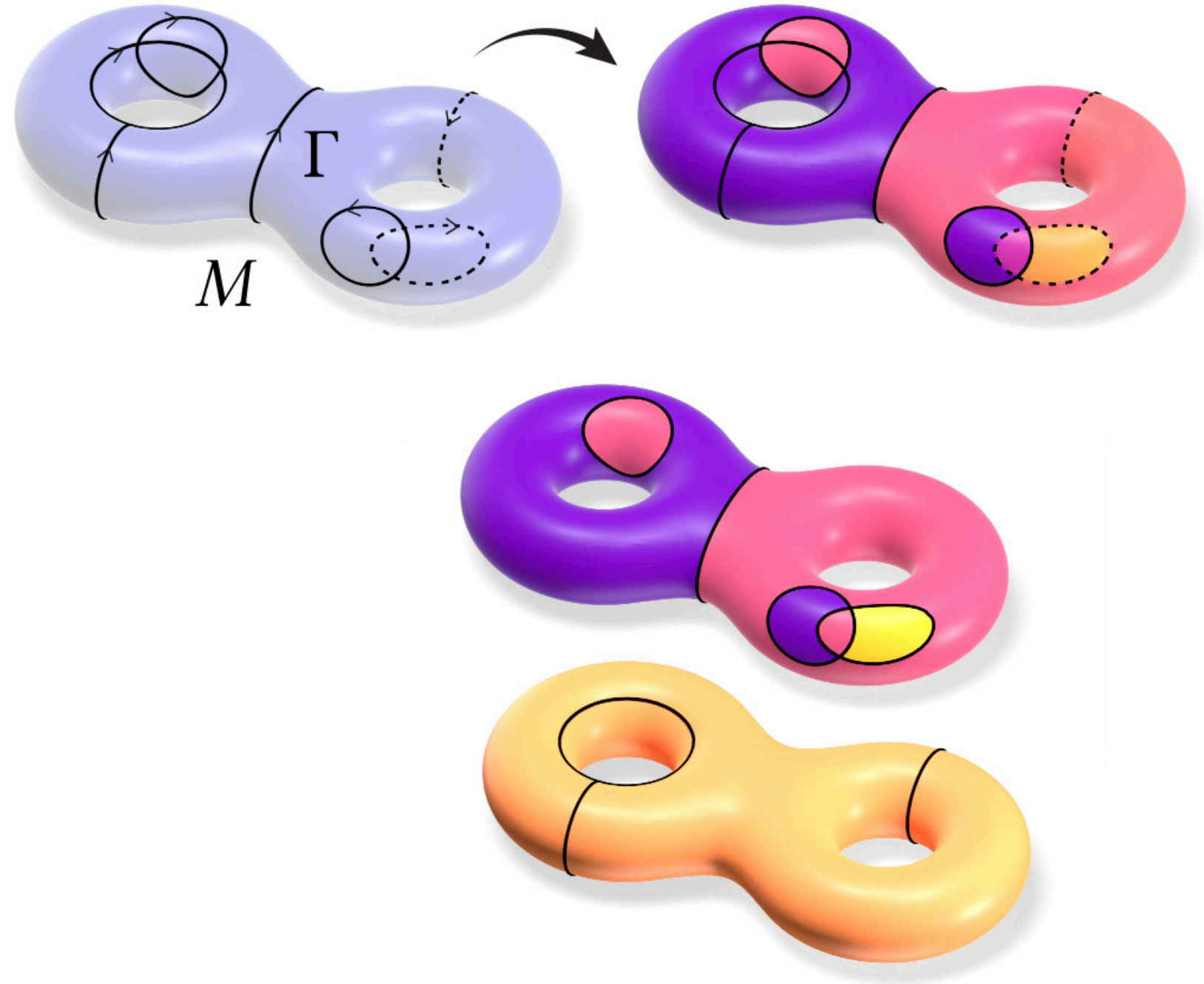
A decomposition of Γ into:

- bounding components that induce valid regions
- nonbounding components.

A closed, completed version of the input curve.

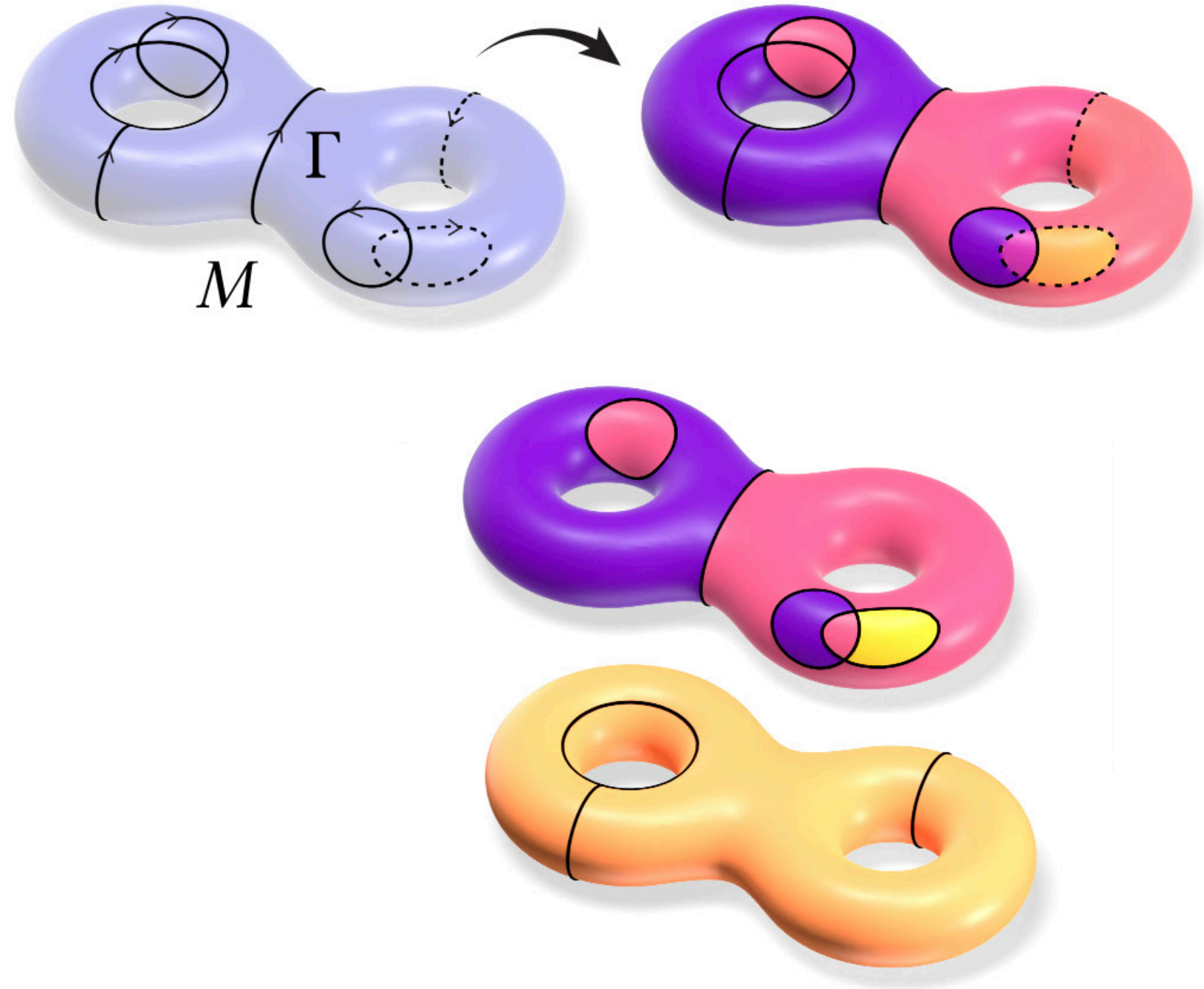


Key insights



Key insights

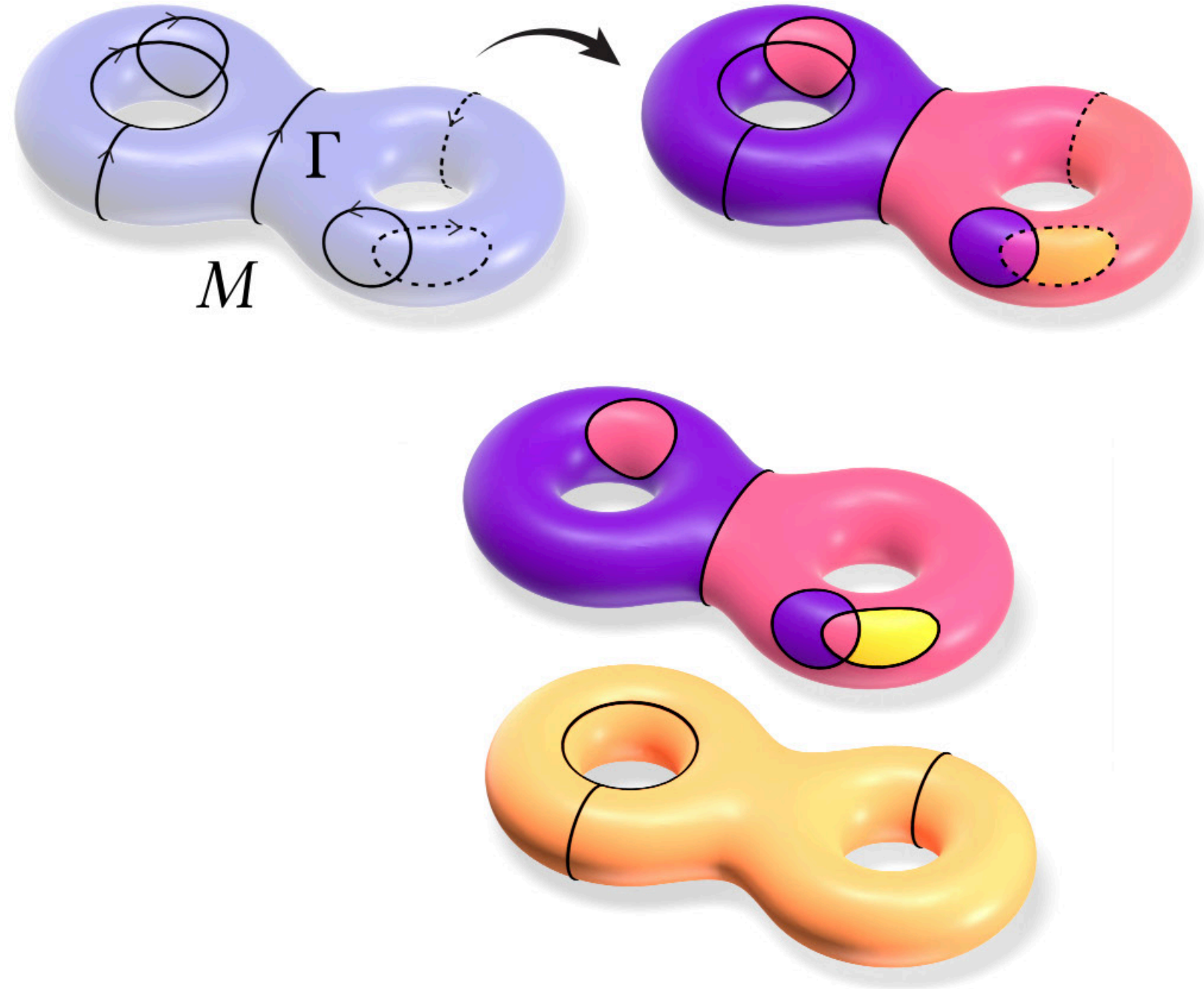
It's difficult to reason about the homology class of *broken* curves.



Key insights

It's difficult to reason about the homology class of *broken* curves.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

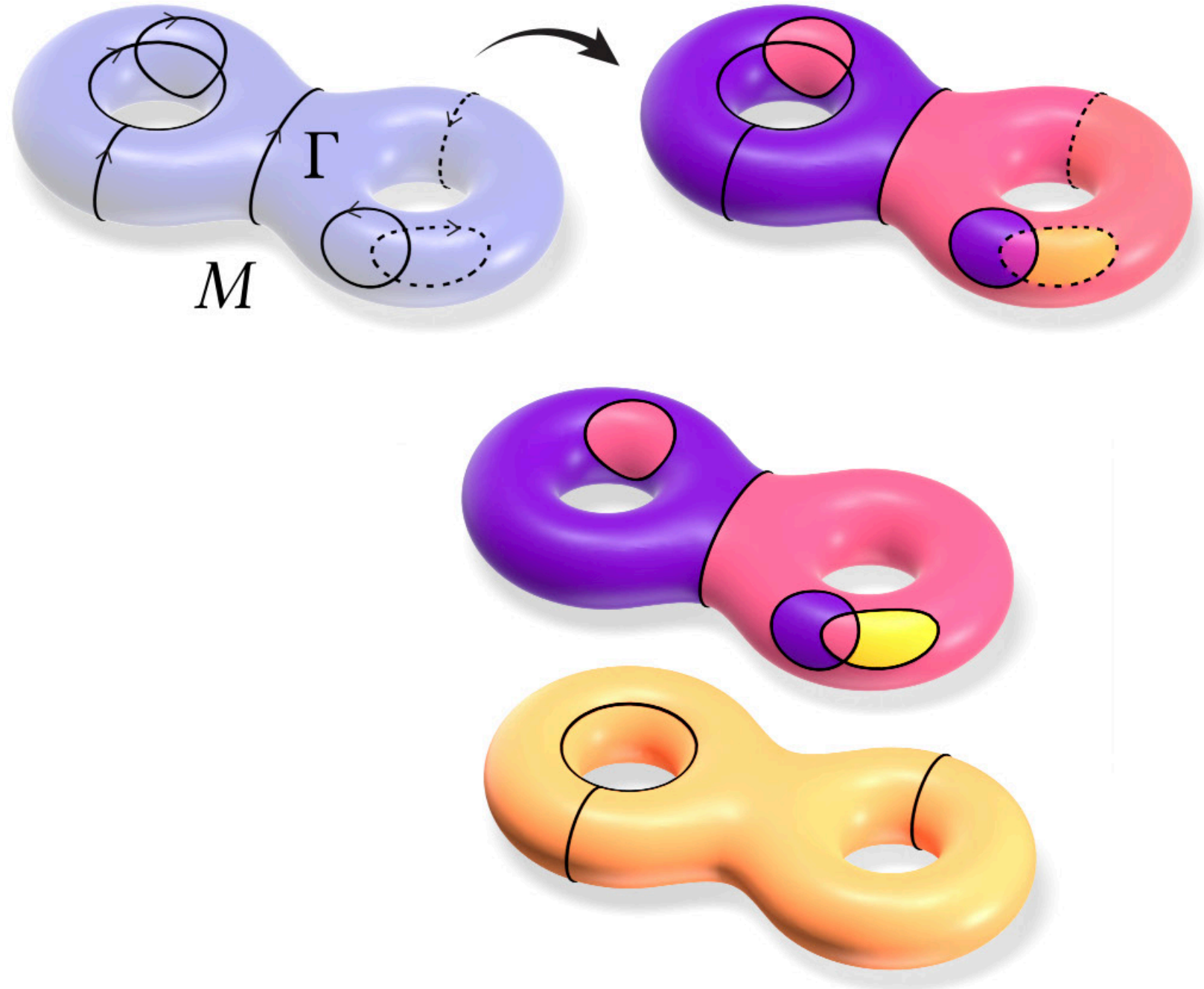


Key insights

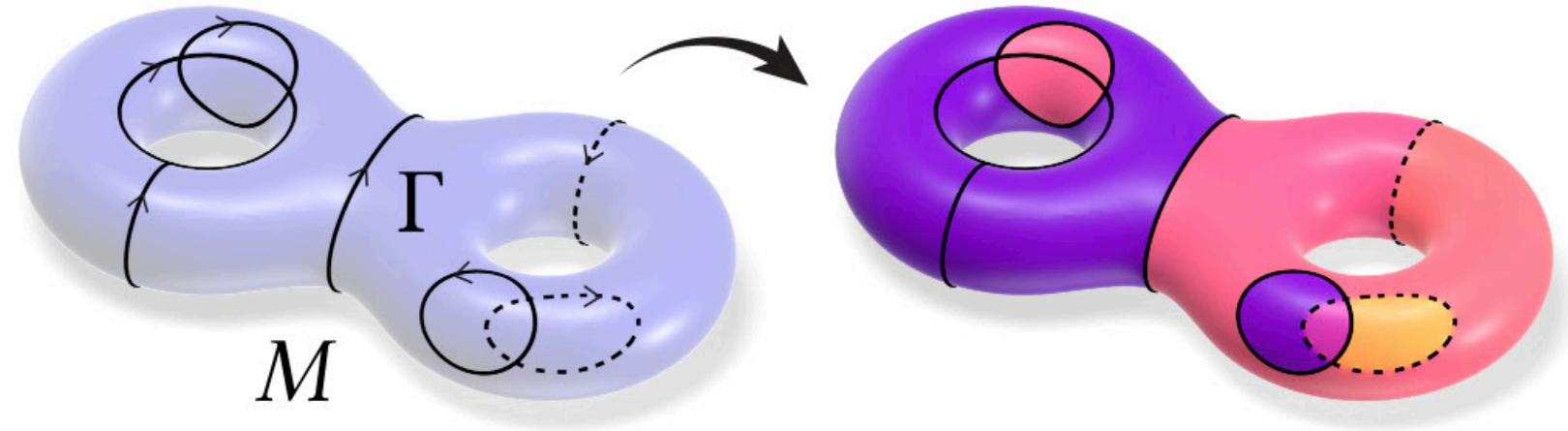
It's difficult to reason about the homology class of *broken* curves.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

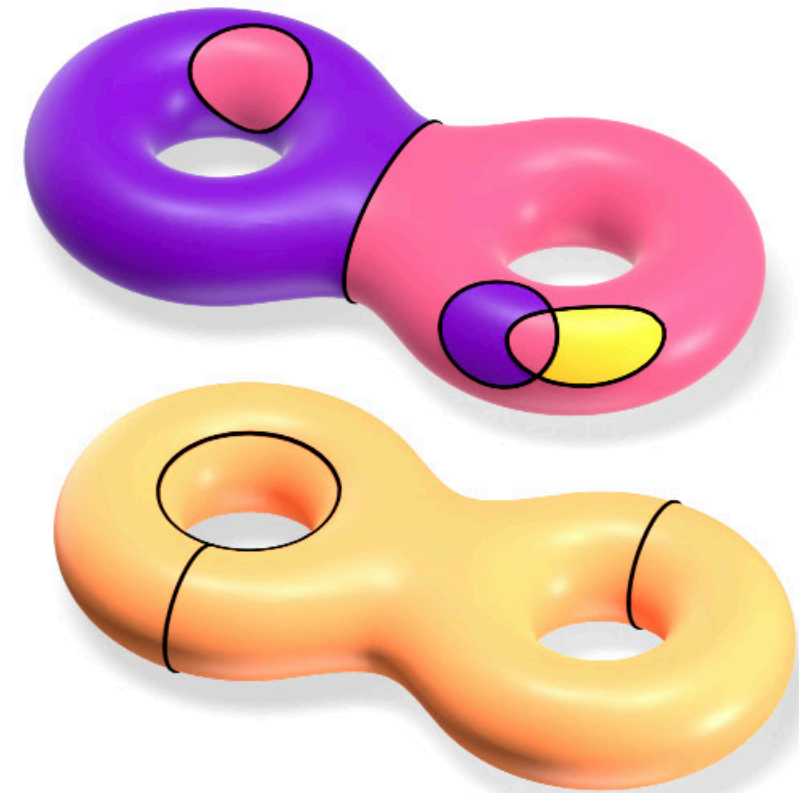
We map from functions back to curves, yielding final output.



Talk outline

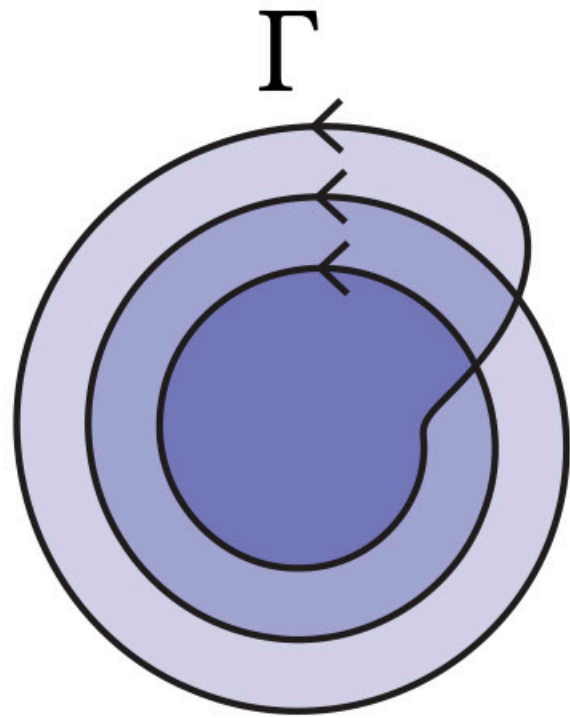


- Algorithm in the smooth setting
- Discretization
- Evaluation & Results

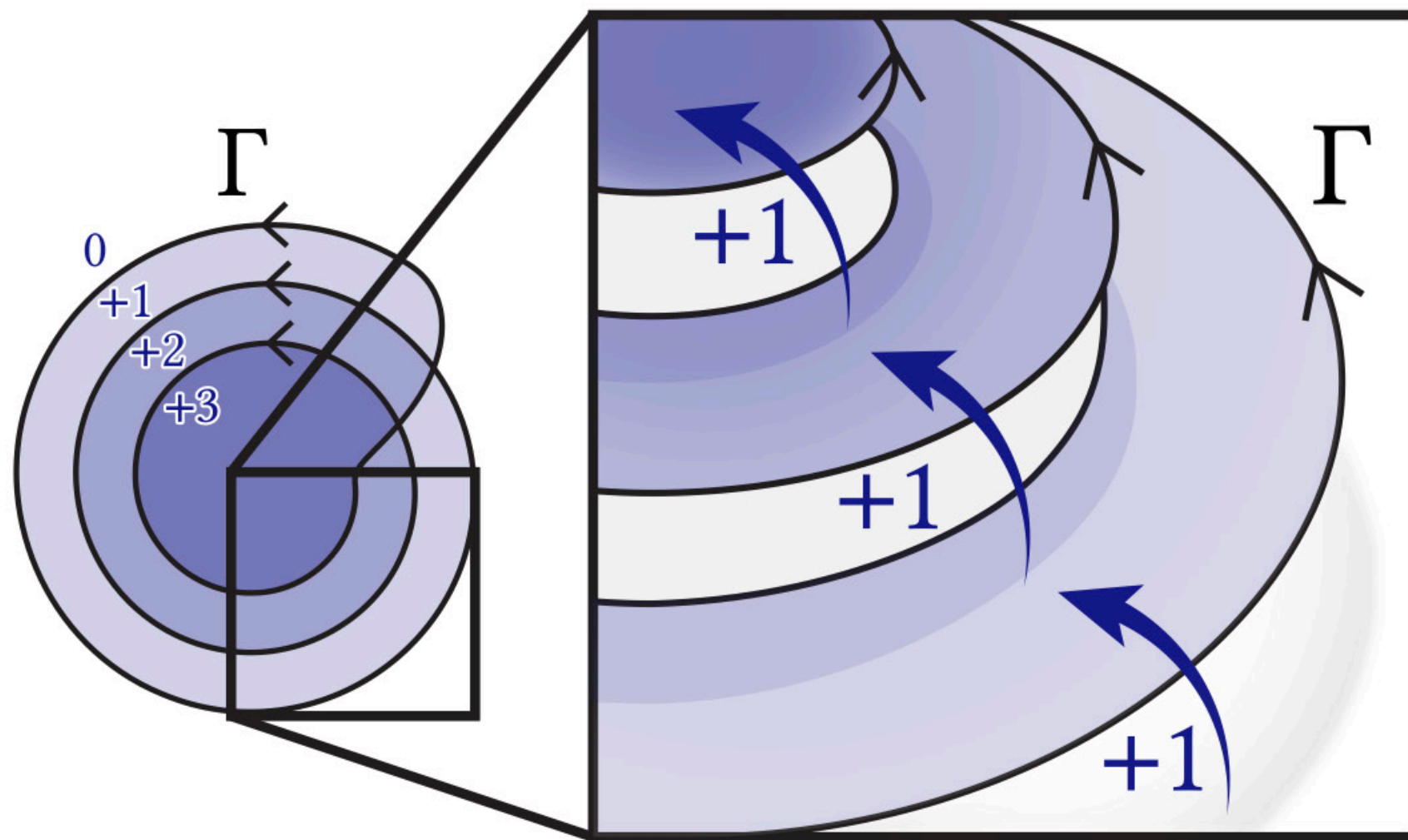


SMOOTH FORMULATION

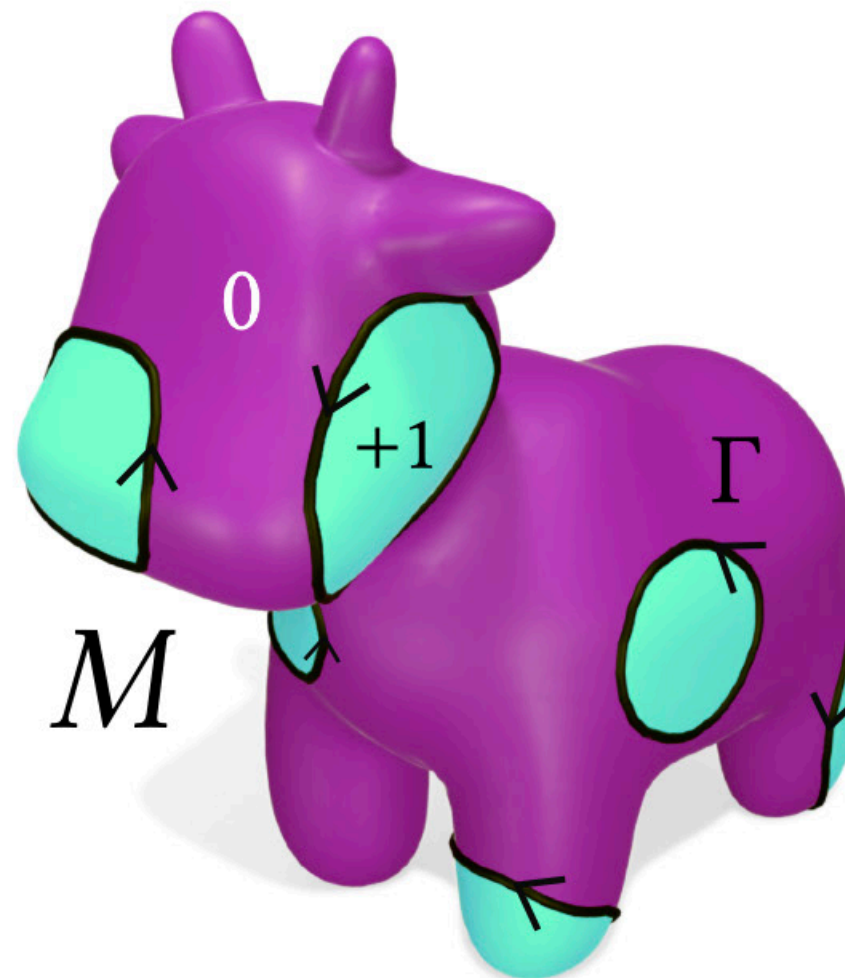
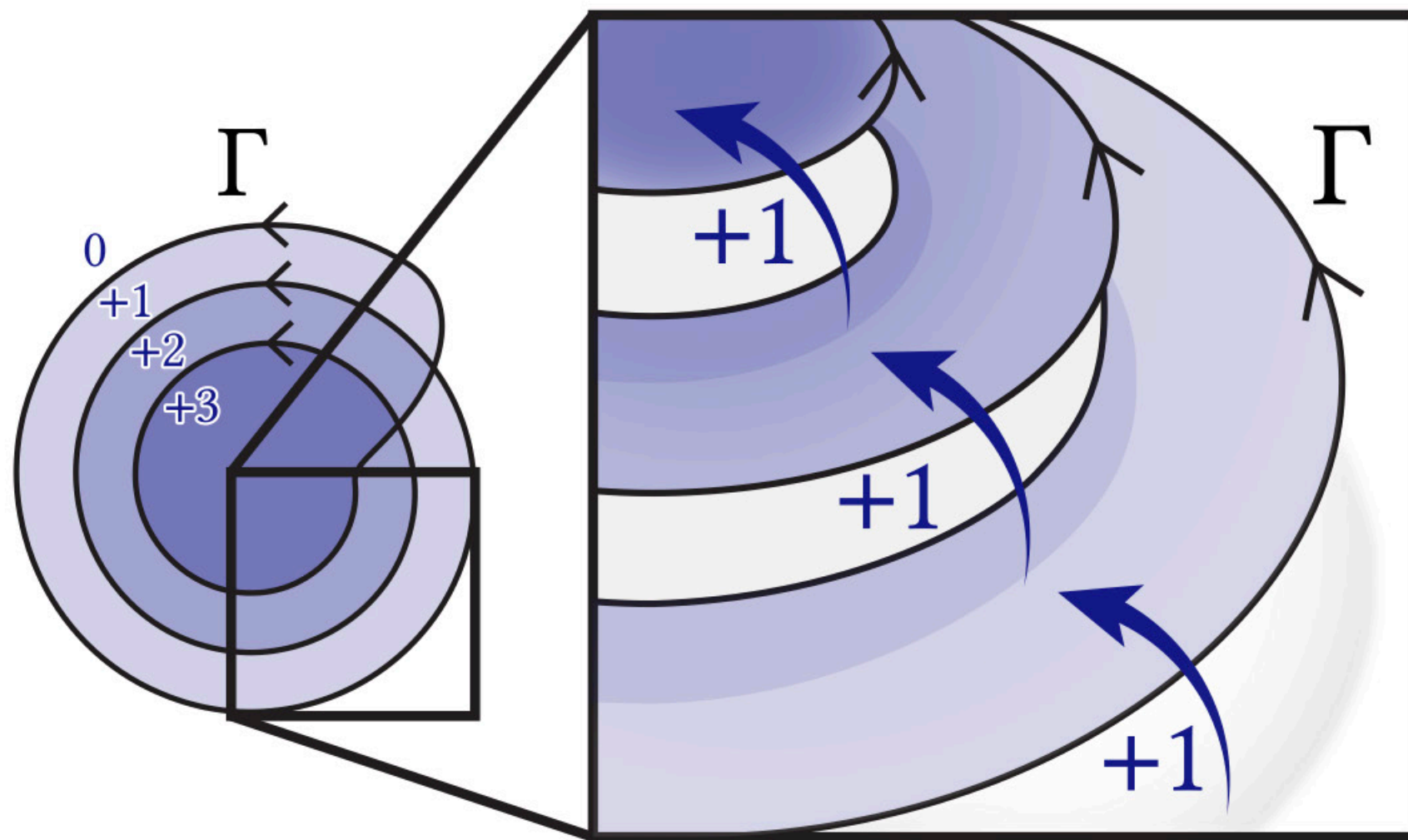
Ordinary winding number: a piecewise constant function



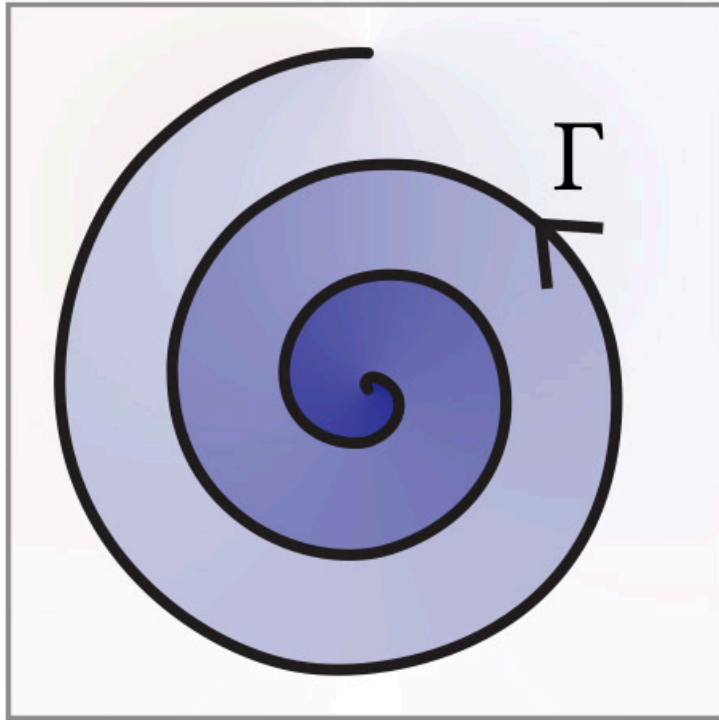
Ordinary winding number: a piecewise constant function



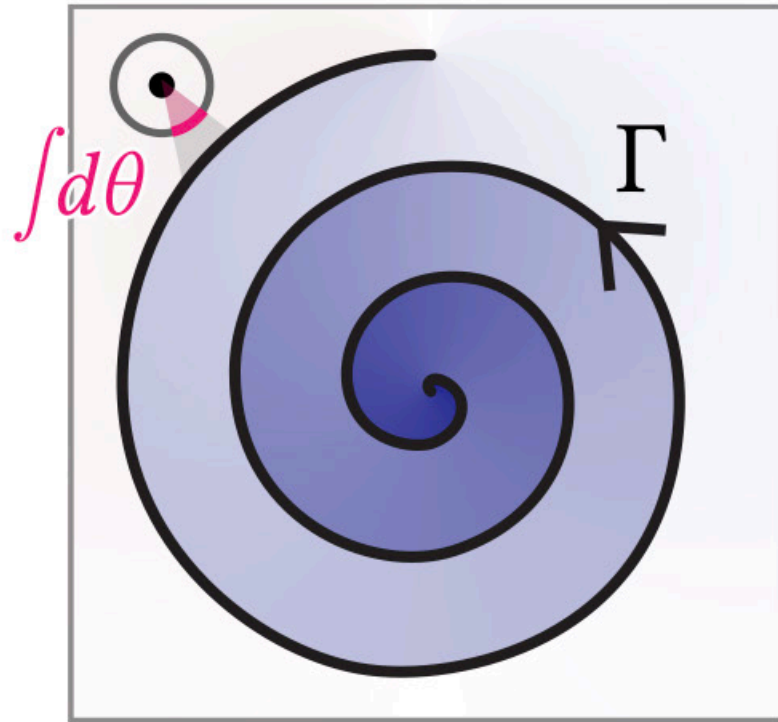
Ordinary winding number: a piecewise constant function



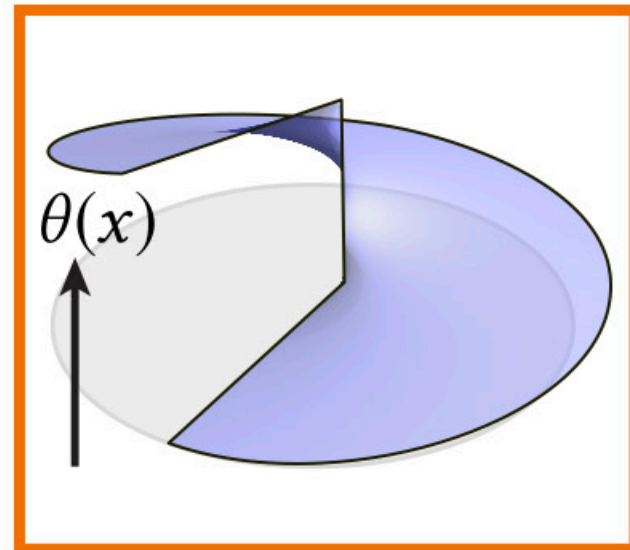
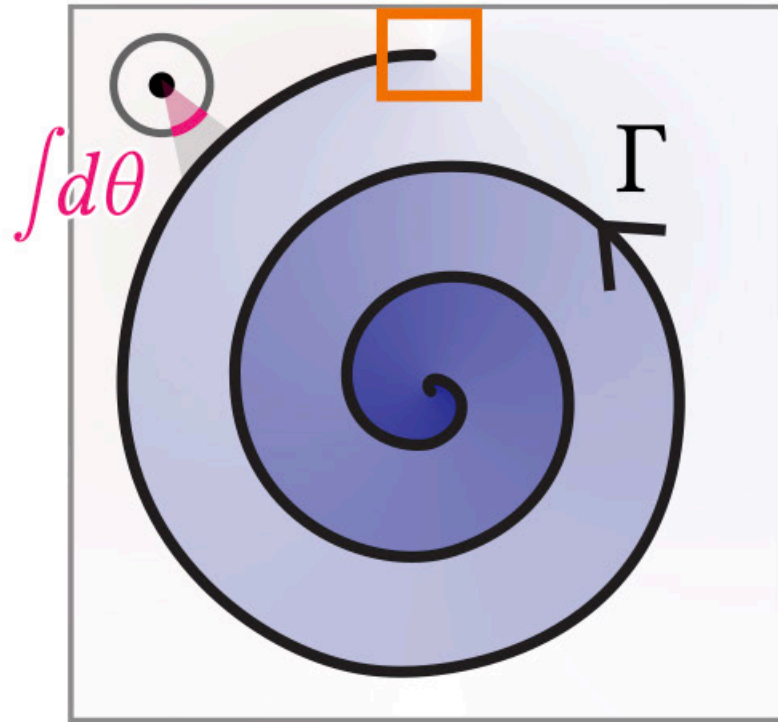
Winding number as a jump harmonic function



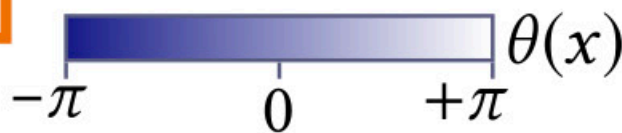
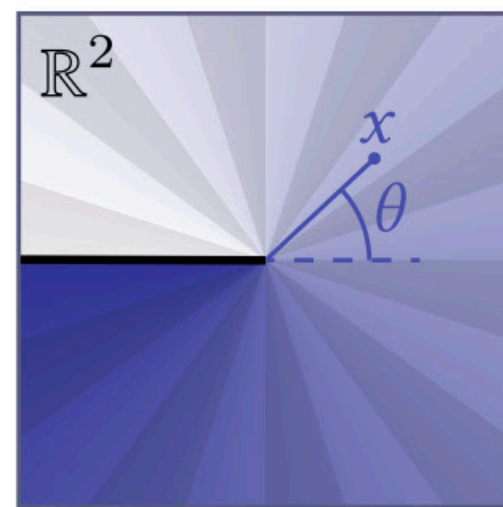
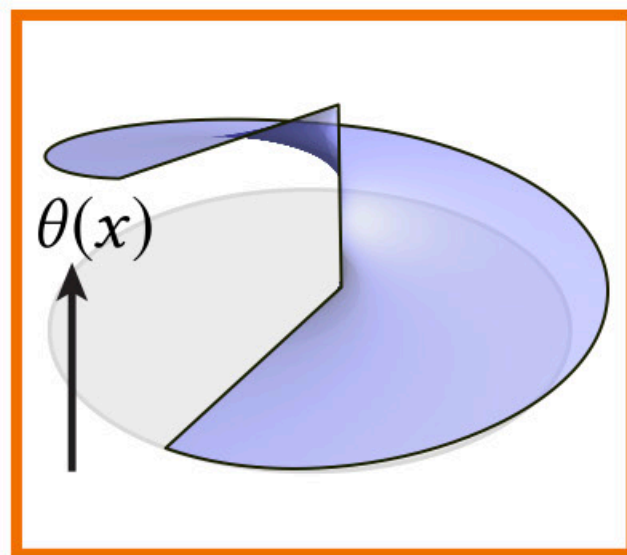
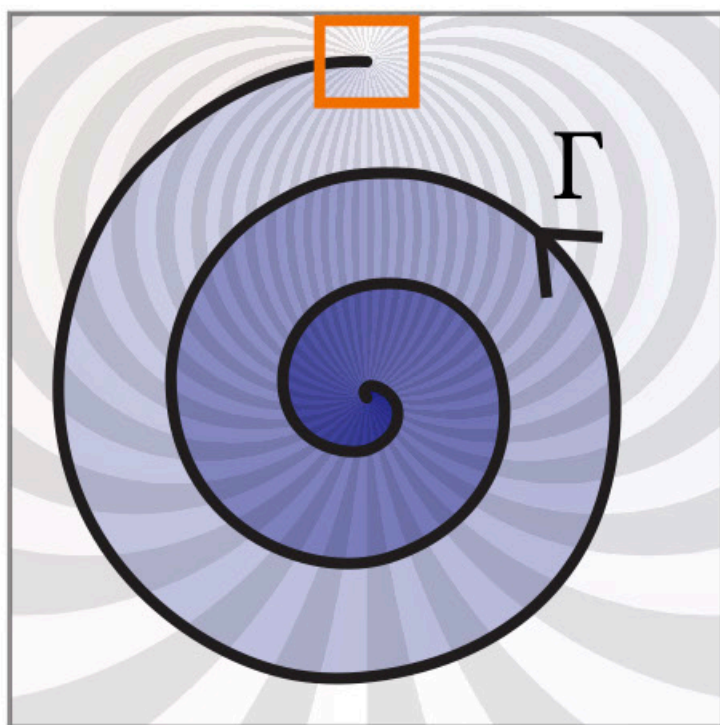
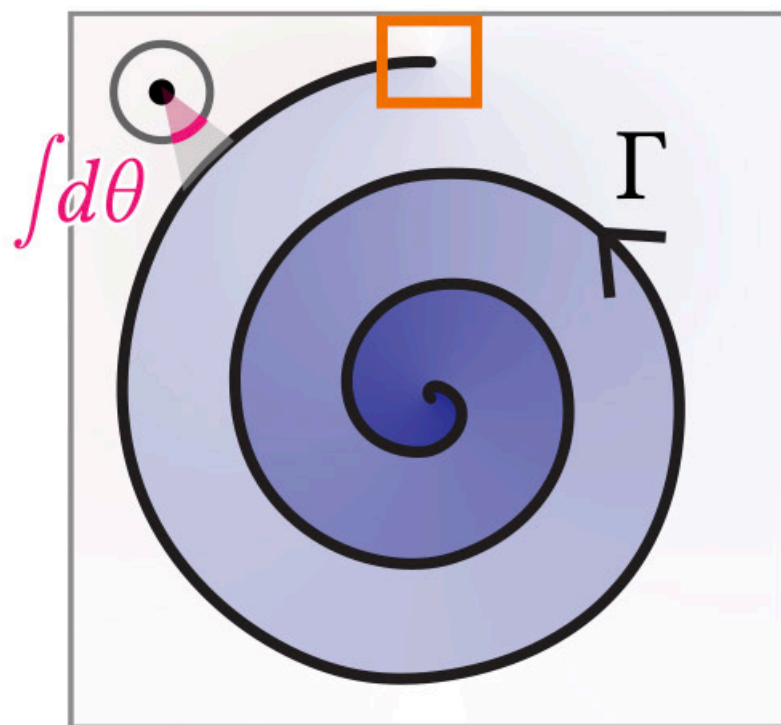
Winding number as a jump harmonic function



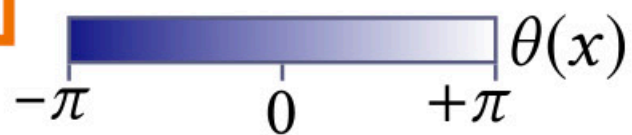
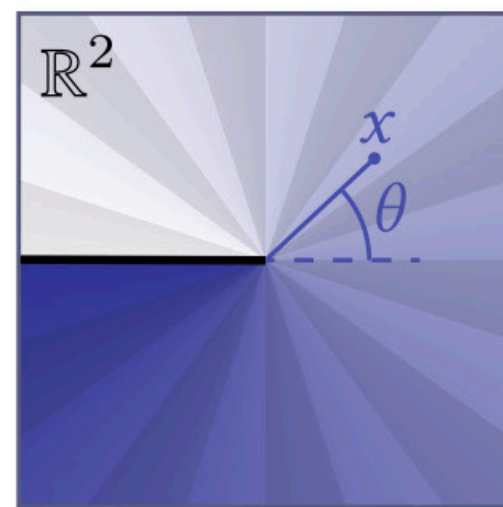
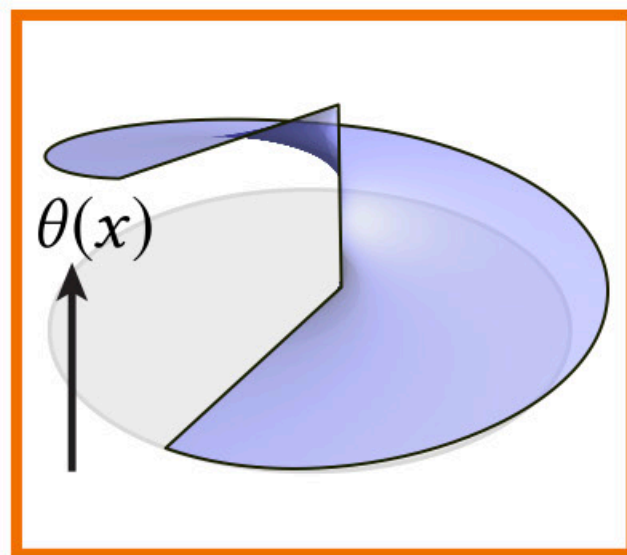
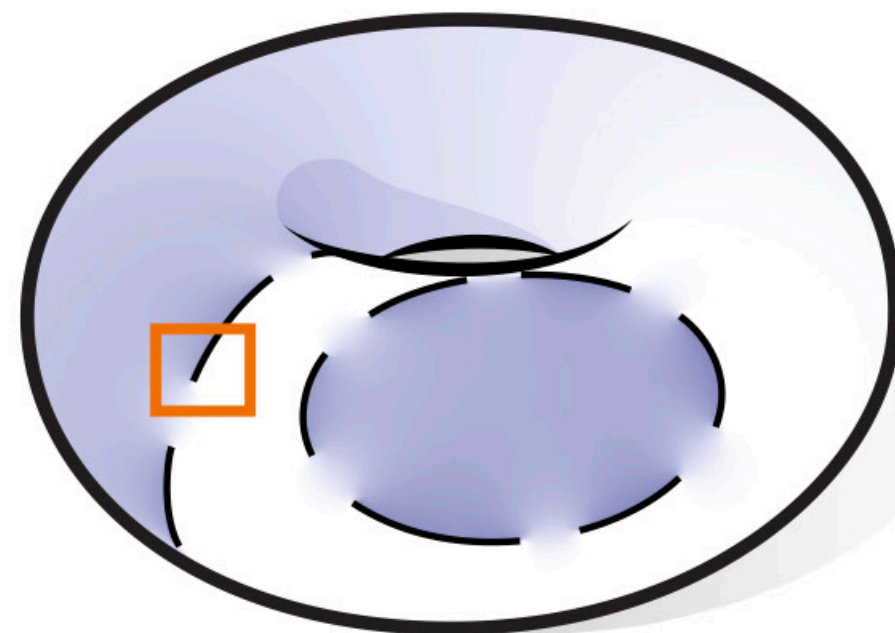
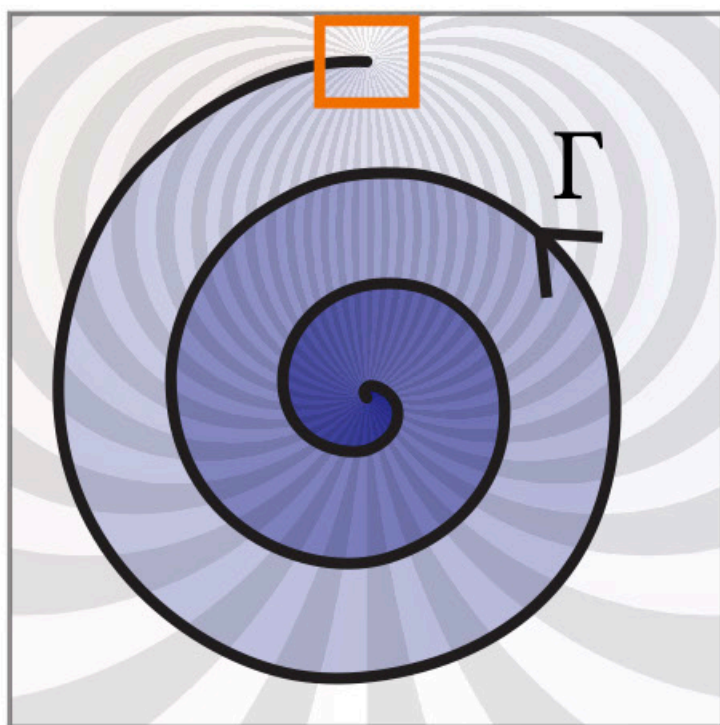
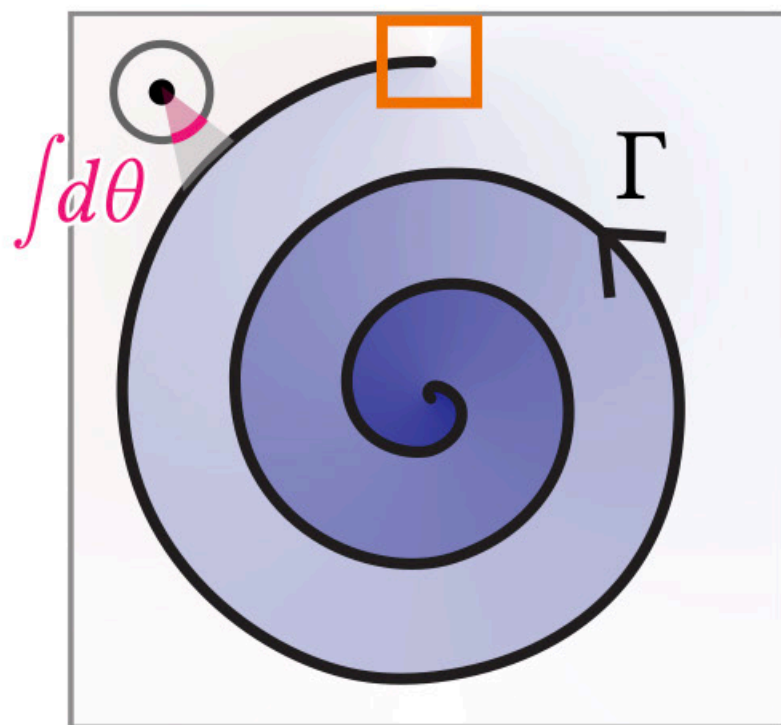
Winding number as a jump harmonic function



Winding number as an angle-valued function



Winding number as an angle-valued function



Winding number as a jump harmonic function

Winding number as a jump harmonic function

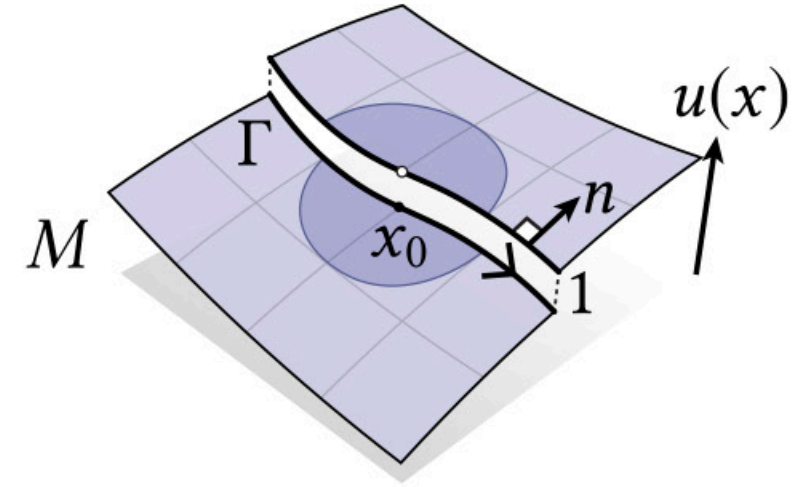
Our starting point is the “Jump Laplace equation”:

$$\Delta u = 0, \quad \text{on } M \setminus \Gamma,$$

Winding number as a jump harmonic function

Our starting point is the “Jump Laplace equation”:

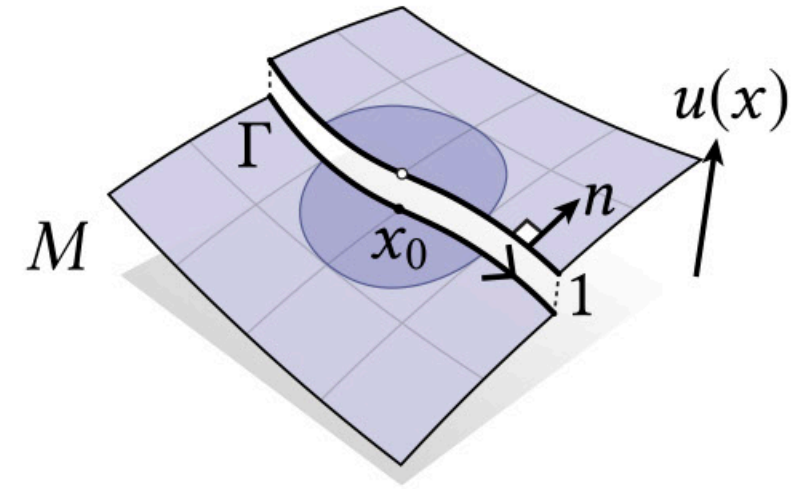
$$\begin{aligned} \Delta u &= 0, & \text{on } M \setminus \Gamma, \\ u^+ - u^- &= 1,^* & \text{on } \Gamma, \end{aligned}$$



Winding number as a jump harmonic function

Our starting point is the “Jump Laplace equation”:

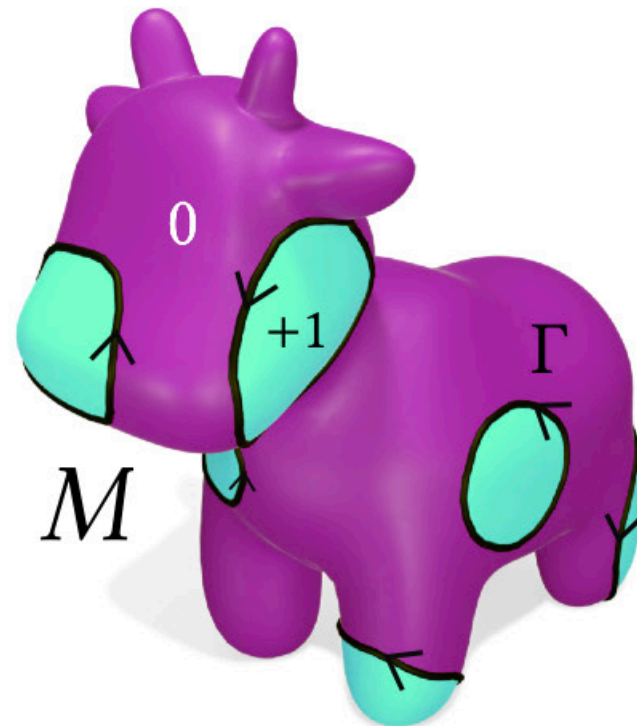
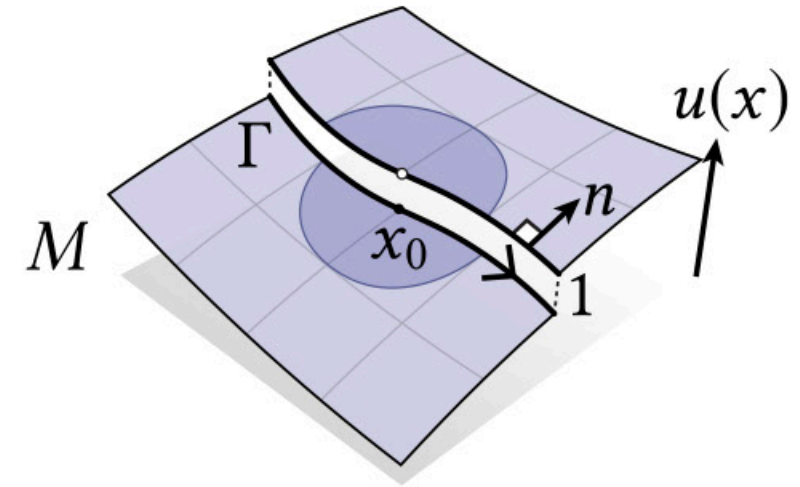
$$\begin{aligned}\Delta u &= 0, && \text{on } M \setminus \Gamma, \\ u^+ - u^- &= 1^*, && \text{on } \Gamma, \\ \partial u^+ / \partial n &= \partial u^- / \partial n, && \text{on } \Gamma.\end{aligned}$$



Winding number as a jump harmonic function

Our starting point is the “Jump Laplace equation”:

$$\begin{aligned}\Delta u &= 0, && \text{on } M \setminus \Gamma, \\ u^+ - u^- &= 1^*, && \text{on } \Gamma, \\ \partial u^+ / \partial n &= \partial u^- / \partial n, && \text{on } \Gamma.\end{aligned}$$

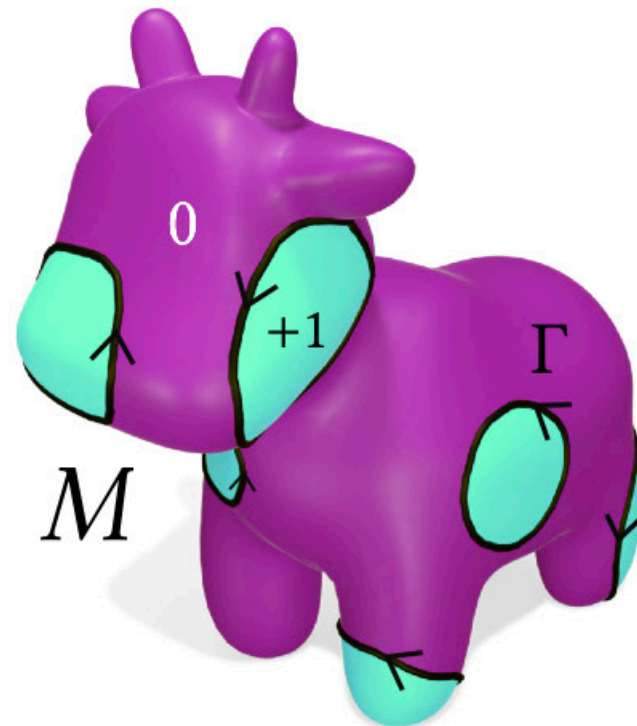
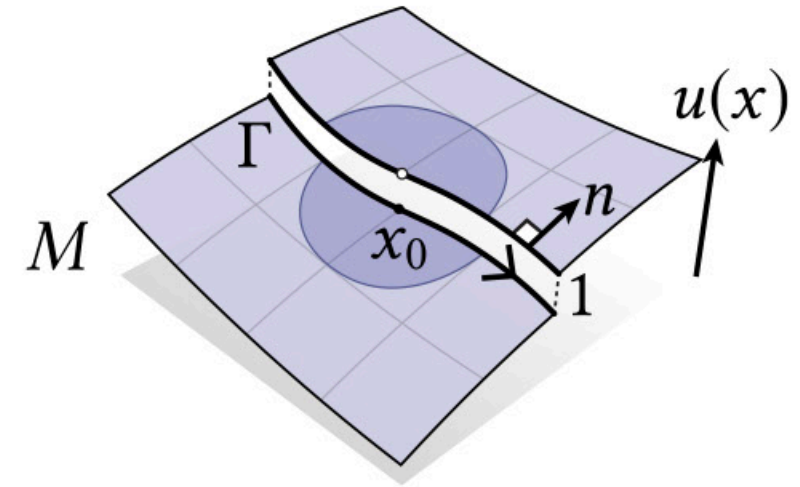


*In general, $u^+(x) - u^-(x) =$ number of times Γ covers x .

Winding number as a jump harmonic function

Our starting point is the “Jump Laplace equation”:

$$\begin{aligned}\Delta u &= 0, && \text{on } M \setminus \Gamma, \\ u^+ - u^- &= 1^*, && \text{on } \Gamma, \\ \partial u^+ / \partial n &= \partial u^- / \partial n, && \text{on } \Gamma.\end{aligned}$$

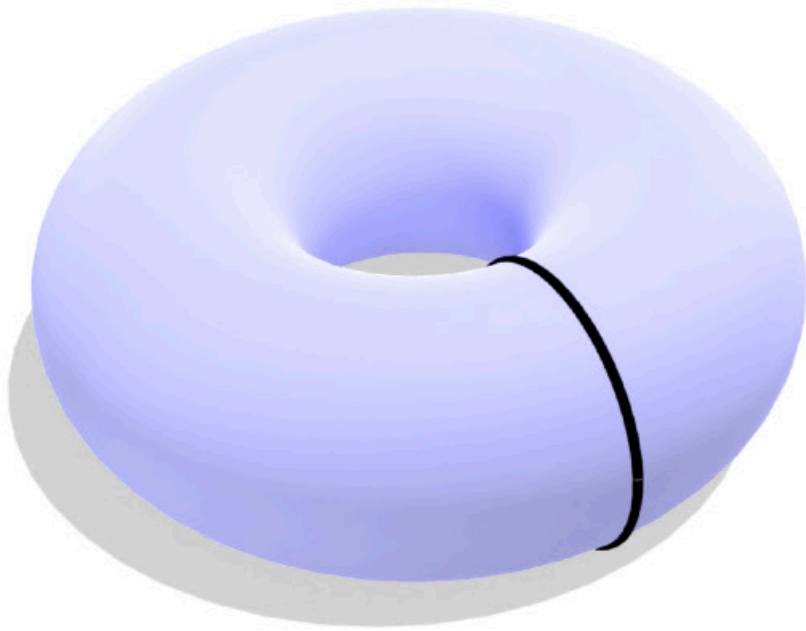


If M is simply-connected,
and all curves are closed,
then we're done!

*In general, $u^+(x) - u^-(x) =$ number of times Γ covers x .

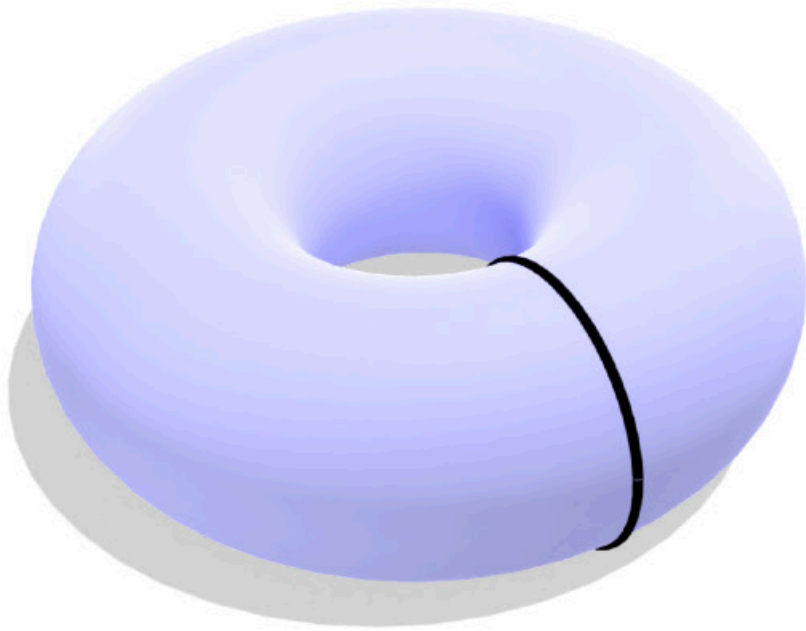
Nonbounding loops complicate region labeling

input

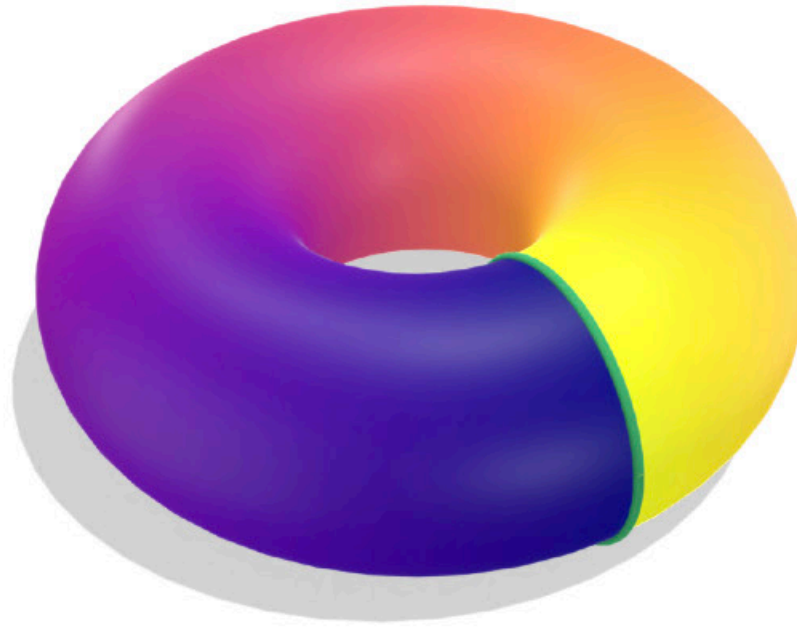


Nonbounding loops complicate region labeling

input

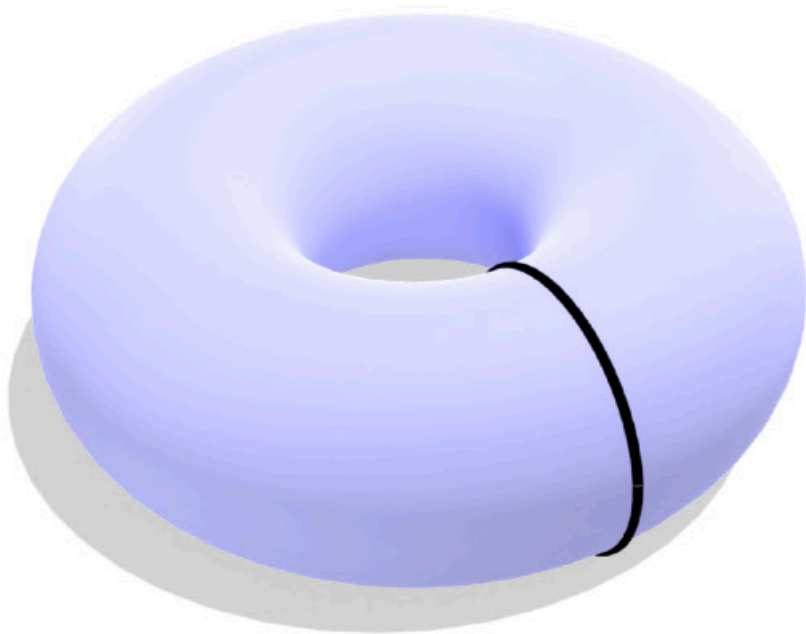


jump harmonic
function u

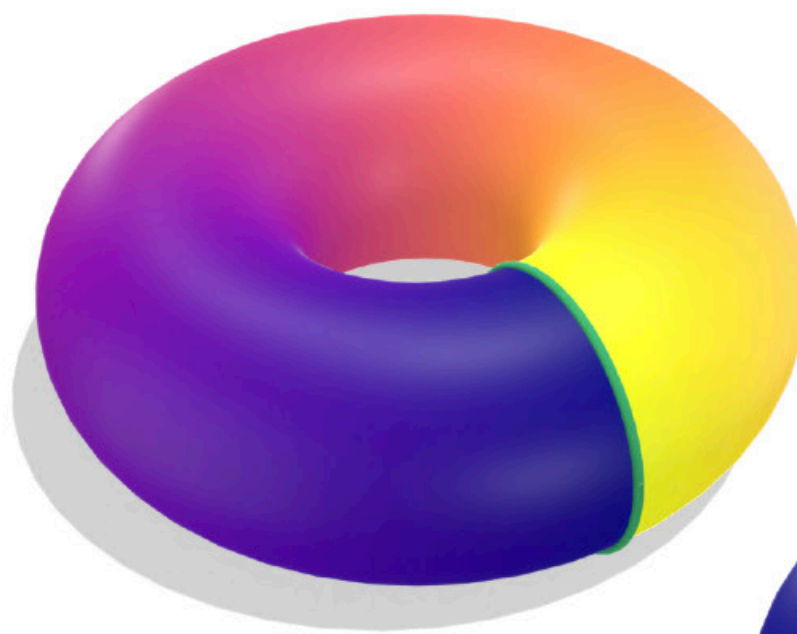


Nonbounding loops complicate region labeling

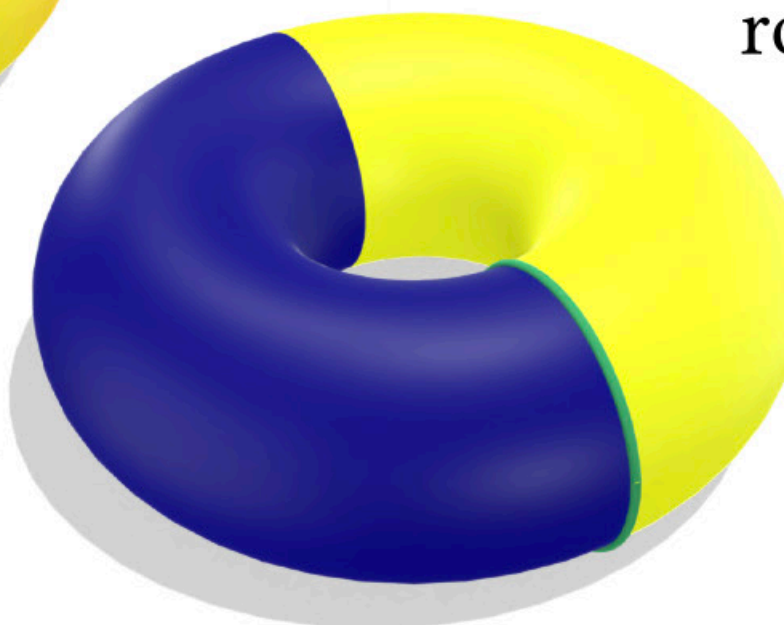
input



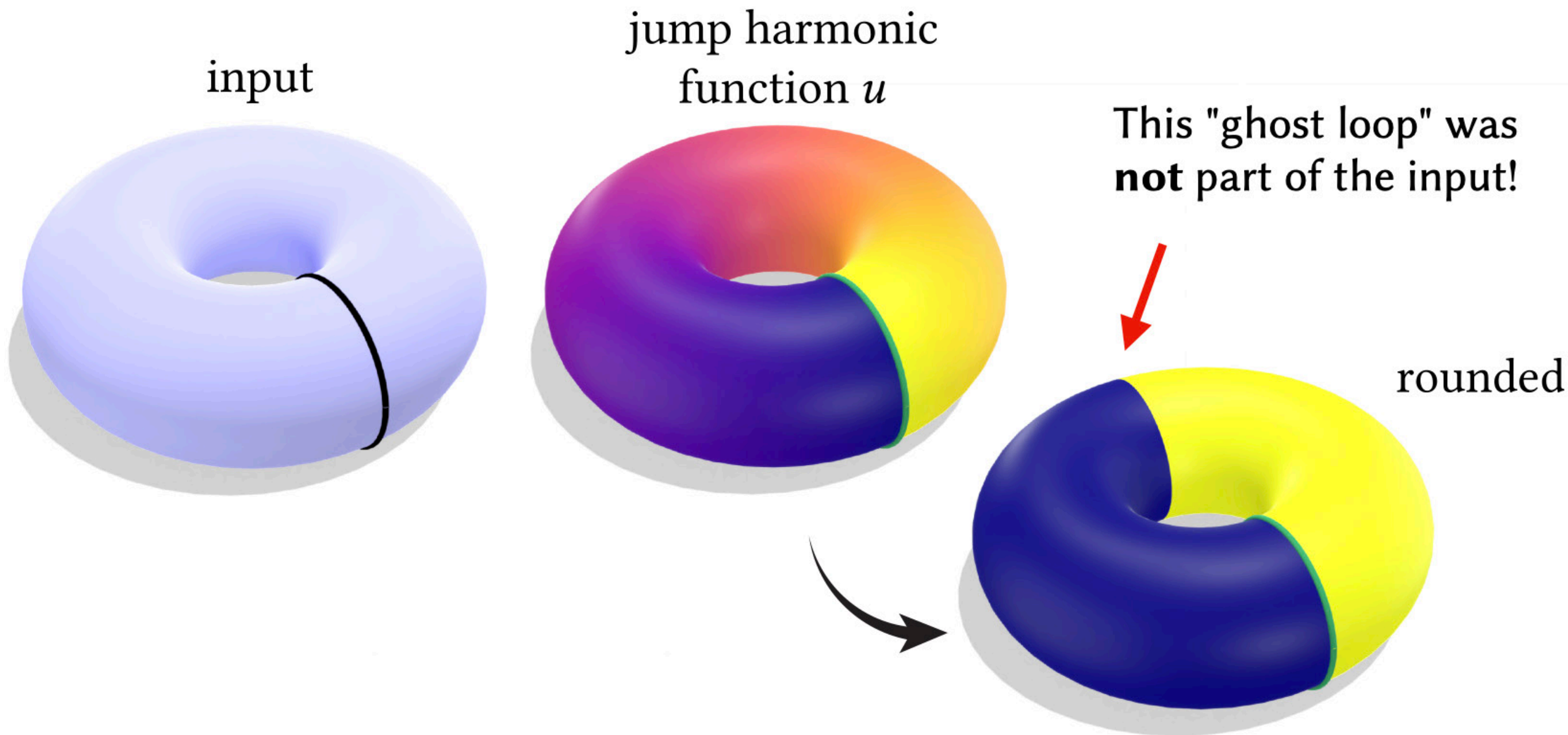
jump harmonic
function u



rounded



Nonbounding loops complicate region labeling



What are nonbounding loops?

What are nonbounding loops?

The *first homology group*

$H_1(M) = \ker(\partial_1) \setminus \text{im}(\partial_2)$ tells us about (closed) curves that are not boundaries of regions.

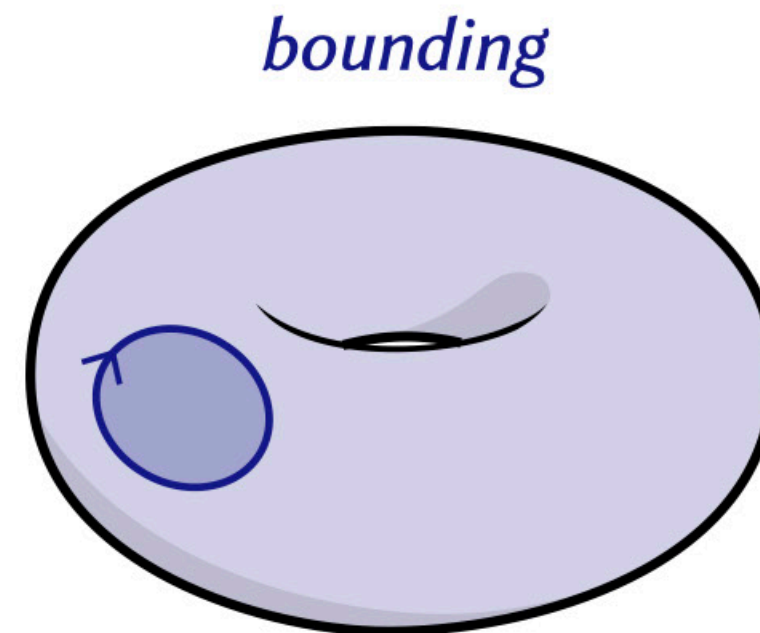
What are nonbounding loops?

The *first homology group*

$H_1(M) = \ker(\partial_1) \setminus \text{im}(\partial_2)$ tells us about (closed) curves that are not boundaries of regions.

For clarity,

bounding := *nullhomologous*



What are nonbounding loops?

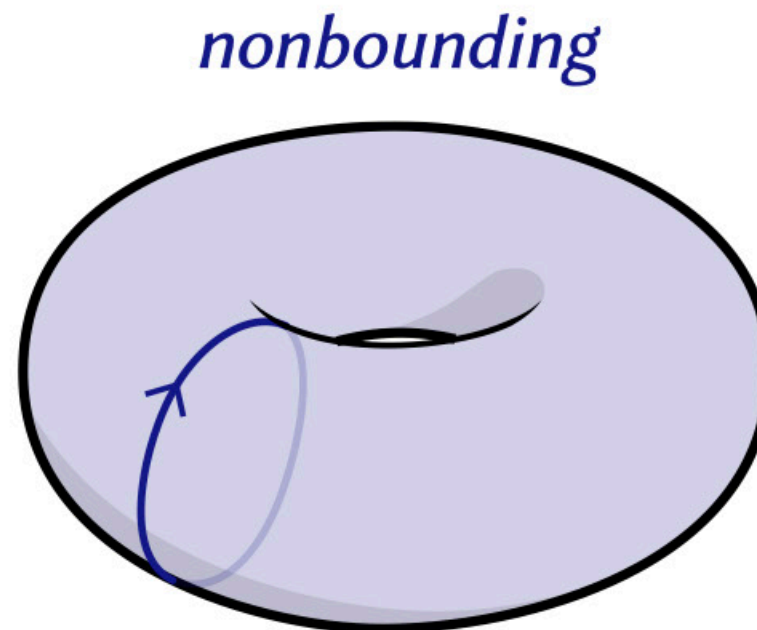
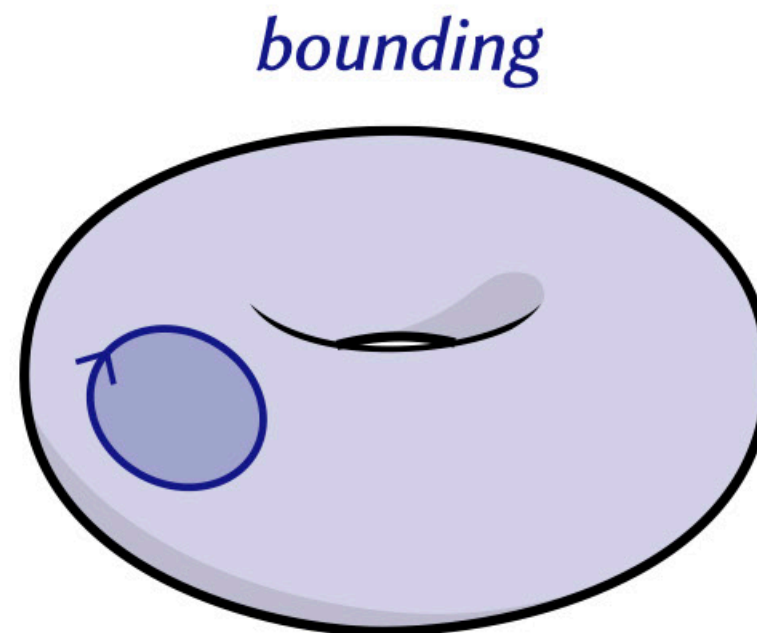
The *first homology group*

$H_1(M) = \ker(\partial_1) \setminus \text{im}(\partial_2)$ tells us about (closed) curves that are not boundaries of regions.

For clarity,

bounding := *nullhomologous*

nonbounding := *non-nullhomologous*



How does homology apply to broken curves?

How does homology apply to broken curves?

It's difficult to reason about curves directly.

How does homology apply to broken curves?

It's difficult to reason about curves directly.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

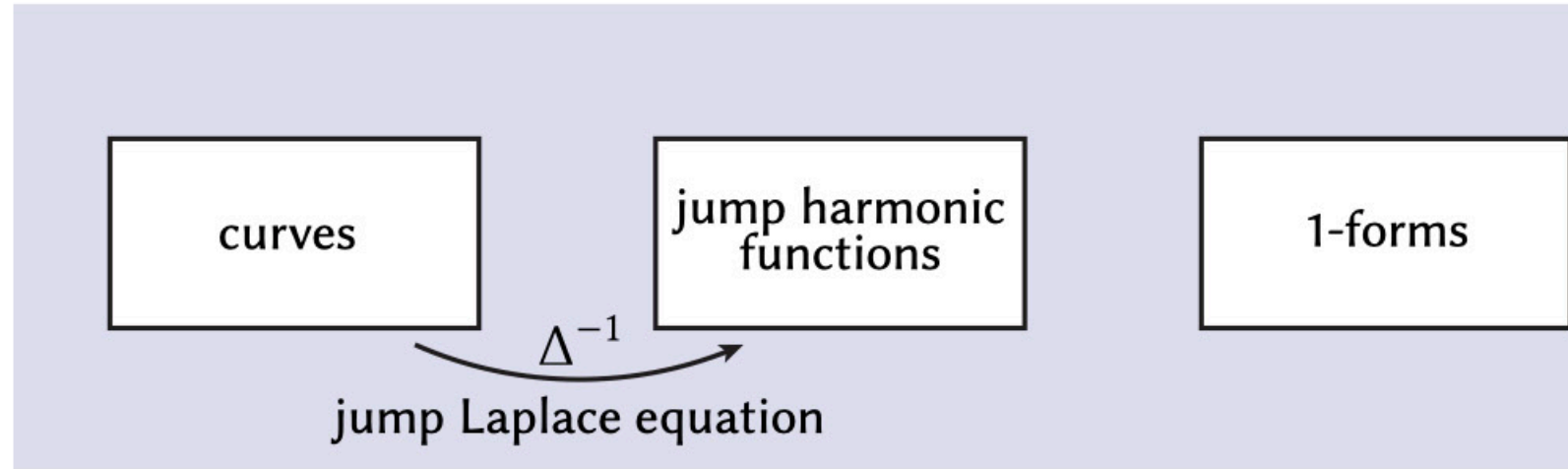
A preview of the journey ahead

curves

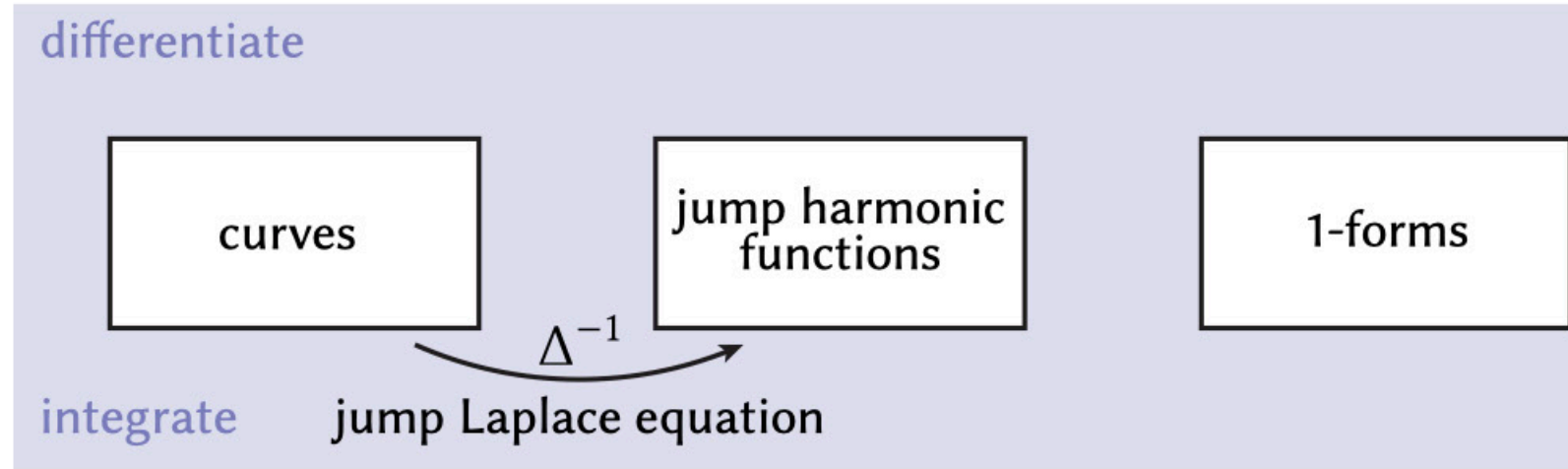
jump harmonic
functions

1-forms

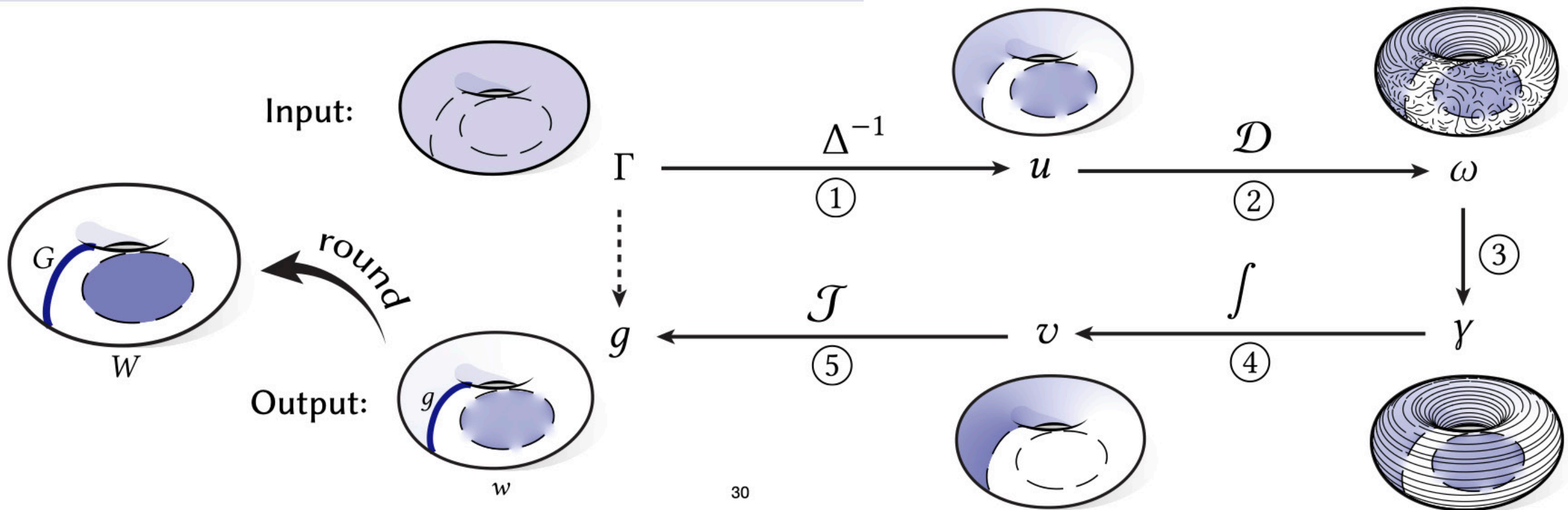
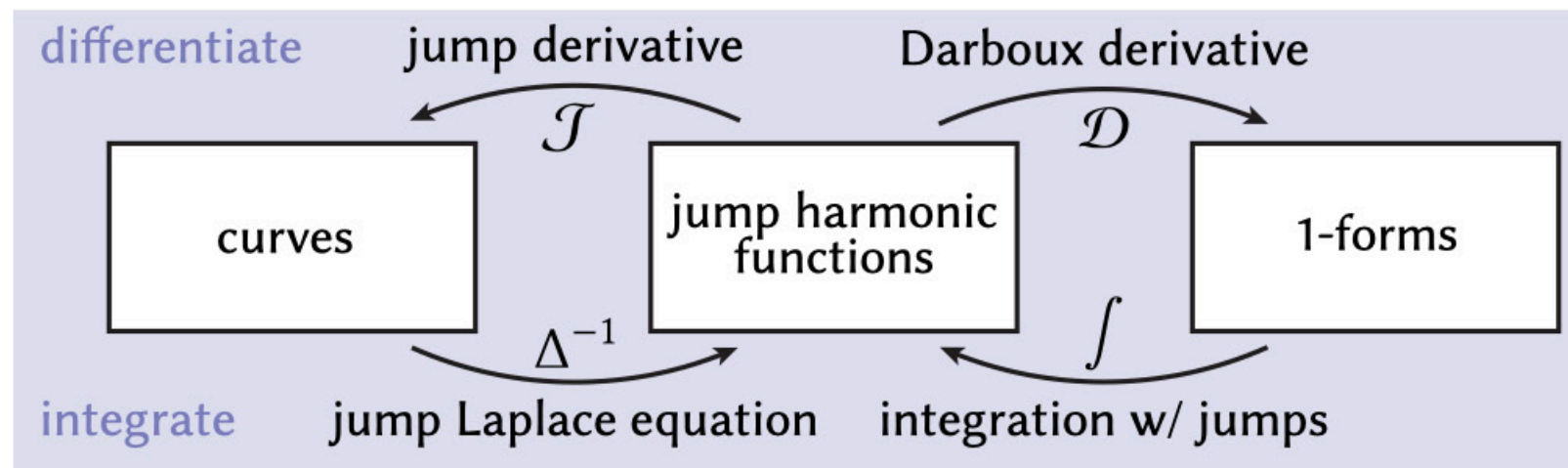
A preview of the journey ahead



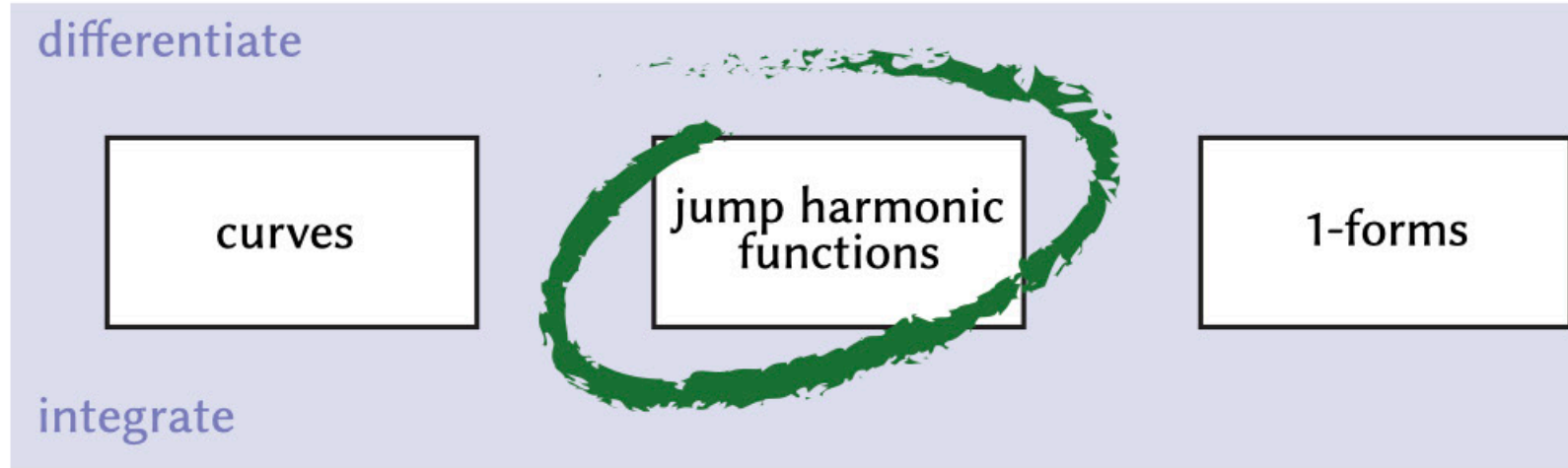
A preview of the journey ahead



A preview of the journey ahead



Jump harmonic functions

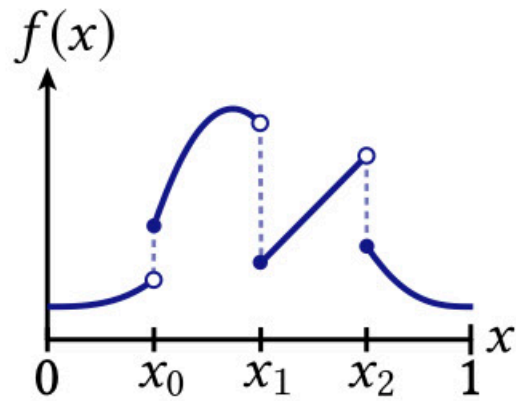


First let's talk about differentiating & integrating jump harmonic functions.

Derivatives of discontinuous functions

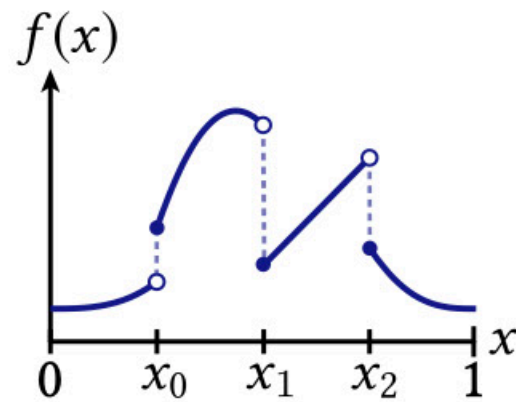
Derivatives of discontinuous functions

Consider a periodic 1D function $f(x)$ on $[0,1]$:



Derivatives of discontinuous functions

Consider a periodic 1D function $f(x)$ on $[0,1]$:

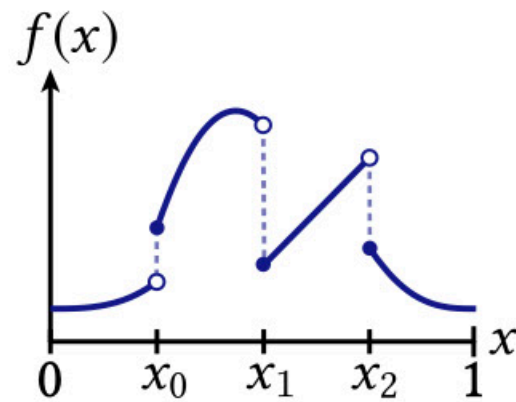


differentiate



Derivatives of discontinuous functions

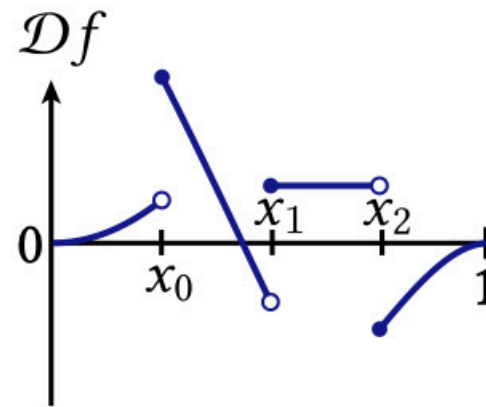
Consider a periodic 1D function $f(x)$ on $[0,1]$:



differentiate

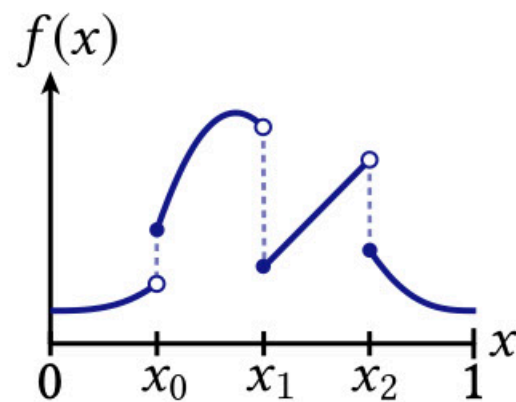


continuous part



Derivatives of discontinuous functions

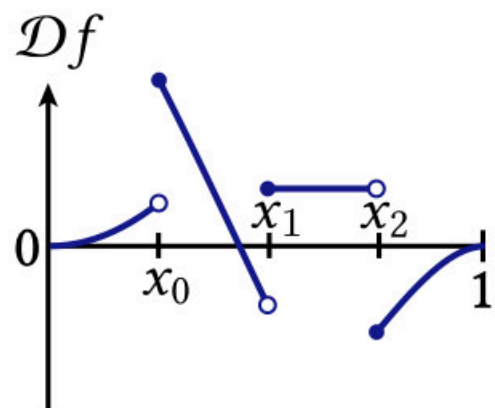
Consider a periodic 1D function $f(x)$ on $[0,1]$:



differentiate



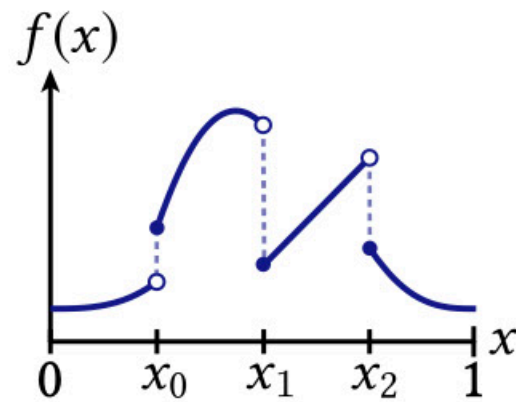
continuous part



$$\omega := \mathcal{D}f$$

Derivatives of discontinuous functions

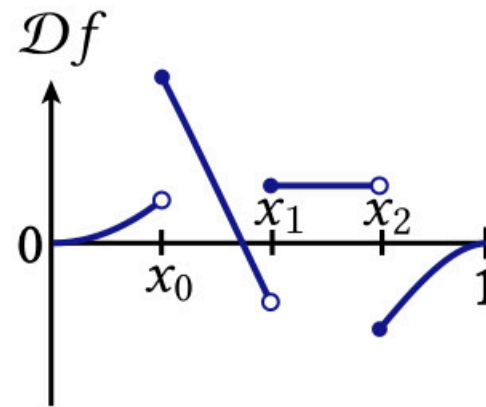
Consider a periodic 1D function $f(x)$ on $[0,1]$:



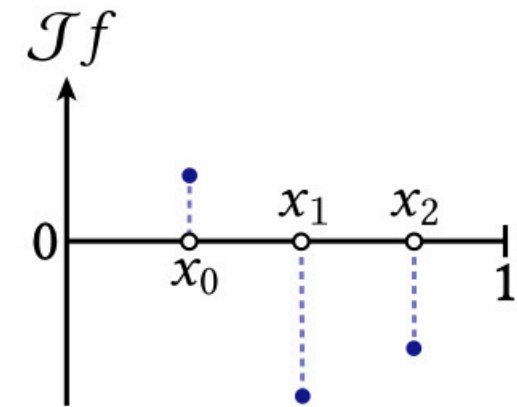
differentiate



continuous part



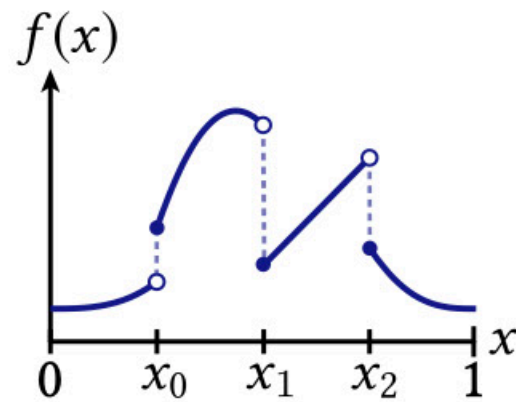
"jump part"



$$\omega := \mathcal{D}f$$

Derivatives of discontinuous functions

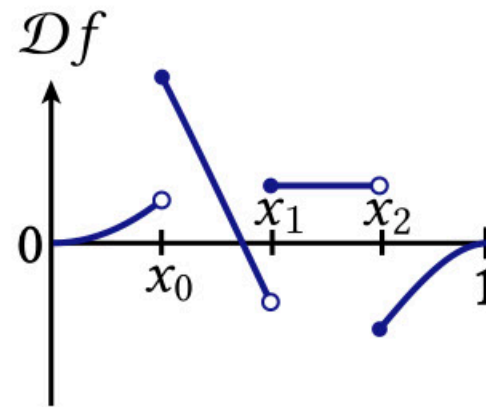
Consider a periodic 1D function $f(x)$ on $[0,1]$:



differentiate



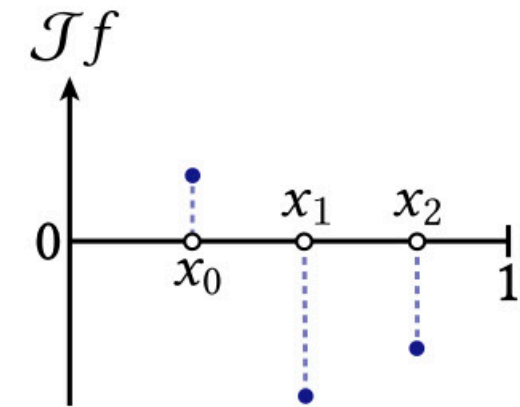
continuous part



$$\omega := \mathcal{D}f$$



"jump part"



$$\mathcal{J}f = \sum_i \Lambda_i \delta_{x_i}$$

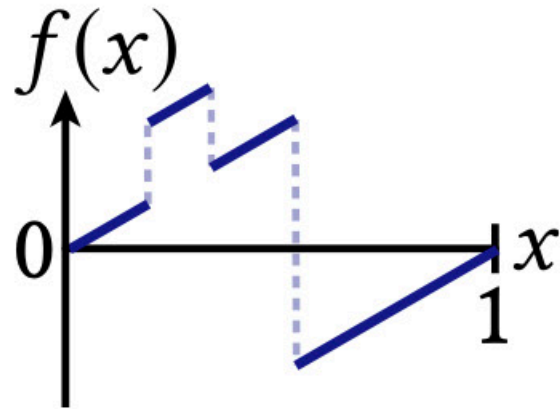
Derivatives of jump harmonic functions

For a **jump harmonic** function f , the *Darboux derivative* $\omega := \mathcal{D}f$ "forgets" jumps:

Derivatives of jump harmonic functions

For a **jump harmonic** function f , the *Darboux derivative* $\omega := \mathcal{D}f$ "forgets" jumps:

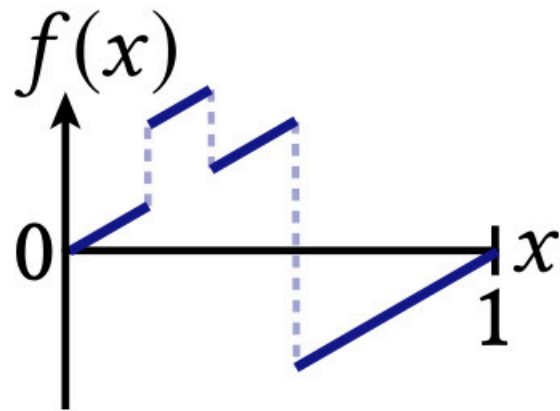
jump harmonic function



Derivatives of jump harmonic functions

For a **jump harmonic** function f , the *Darboux derivative* $\omega := \mathcal{D}f$ "forgets" jumps:

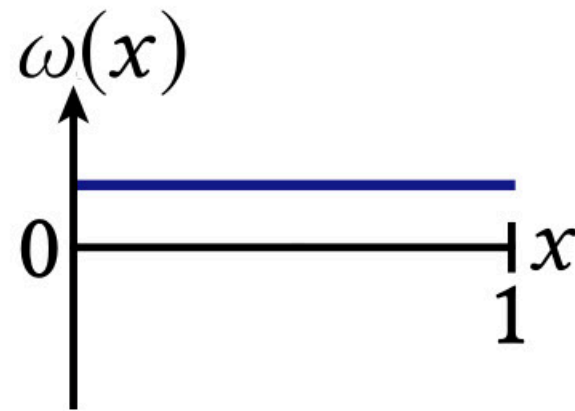
jump harmonic function



differentiate



Darboux derivative

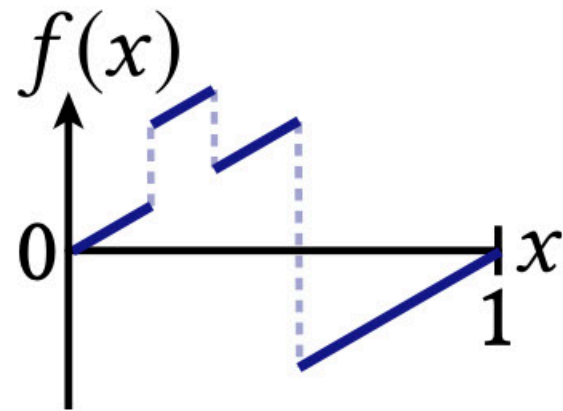


$$\omega := \mathcal{D}f$$

Derivatives of jump harmonic functions

For a **jump harmonic** function f , the *Darboux derivative* $\omega := \mathcal{D}f$ "forgets" jumps:

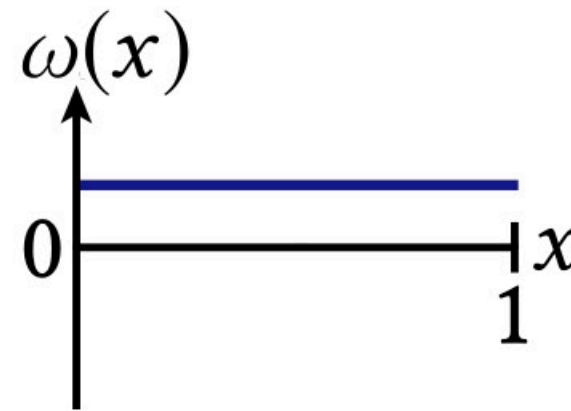
jump harmonic function



differentiate



Darboux derivative



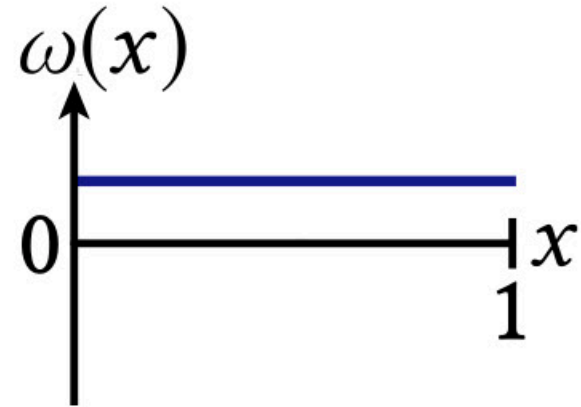
+ jump derivative

$$\omega := \mathcal{D}f$$

Integrating the Darboux derivative

We can only integrate "up to jumps":

Darboux derivative

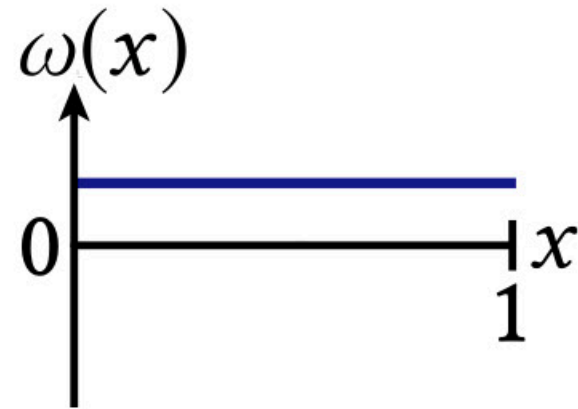


$$\omega := \mathcal{D}f$$

Integrating the Darboux derivative

We can only integrate "up to jumps":

Darboux derivative



integrate

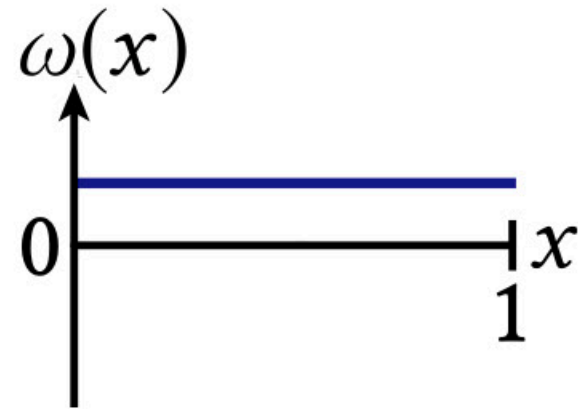


$$\omega := \mathcal{D}f$$

Integrating the Darboux derivative

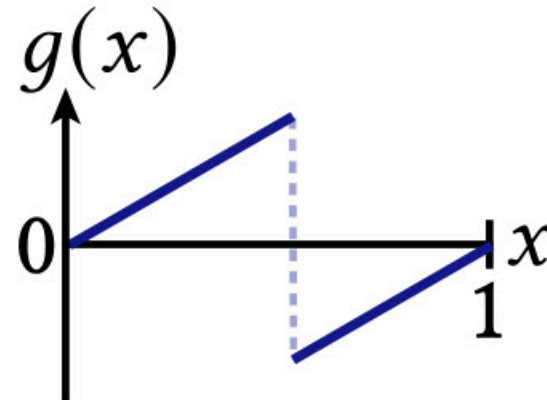
We can only integrate "up to jumps":

Darboux derivative



$$\omega := \mathcal{D}f$$

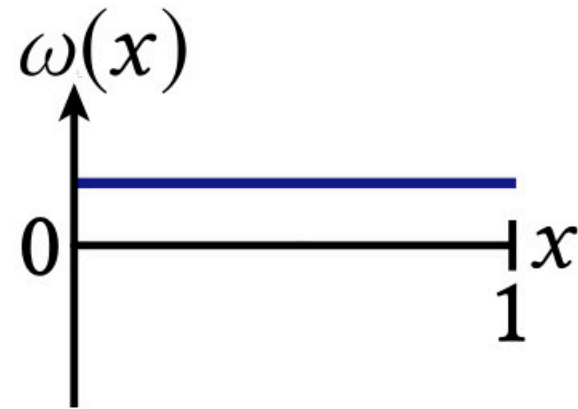
integrate
→



Integrating the Darboux derivative

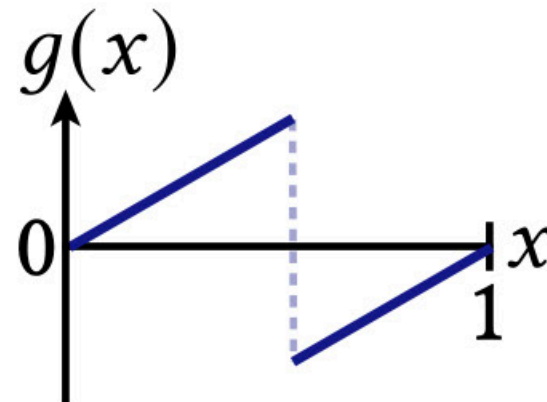
We can only integrate "up to jumps":

Darboux derivative

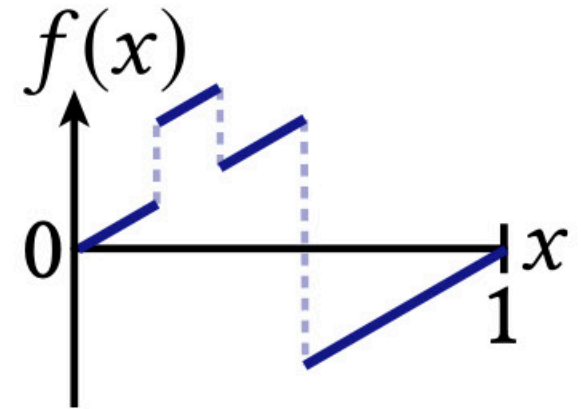


$$\omega := \mathcal{D}f$$

integrate
→



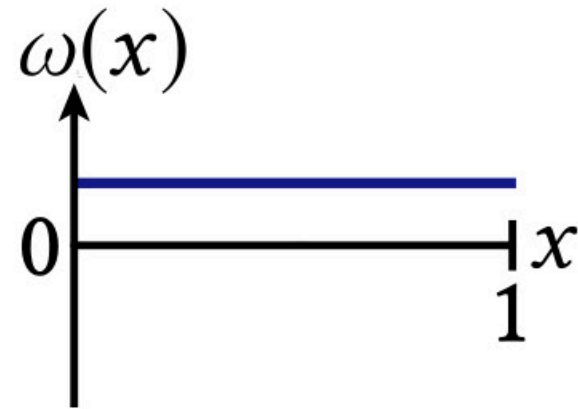
\neq



Integrating the Darboux derivative

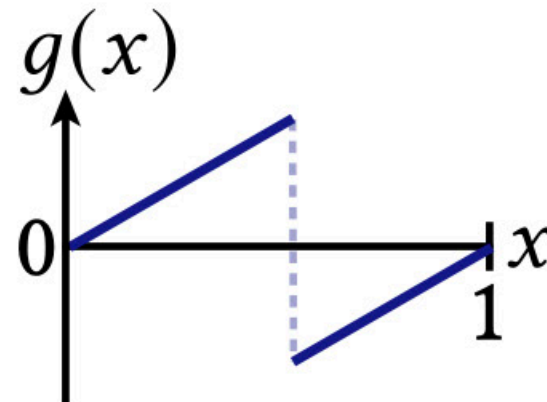
We can only integrate "up to jumps":

Darboux derivative

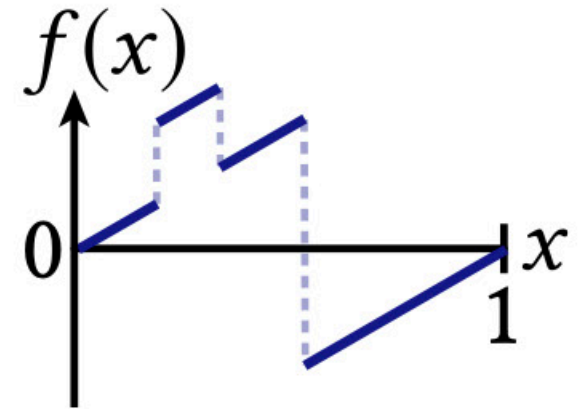


$$\omega := \mathcal{D}f$$

integrate
→



\neq

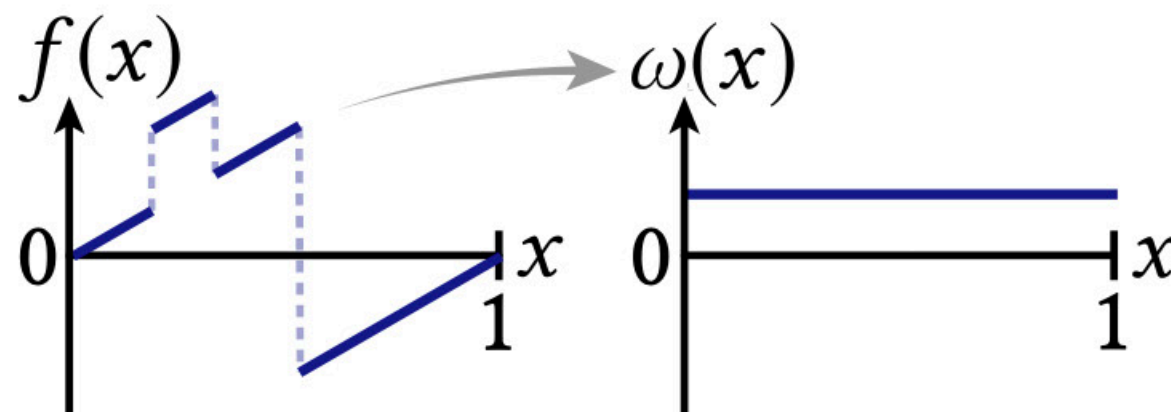


Moral: To map from derivatives back to curves, we can integrate ω – and choose the jumps!

2D jump harmonic functions — same story

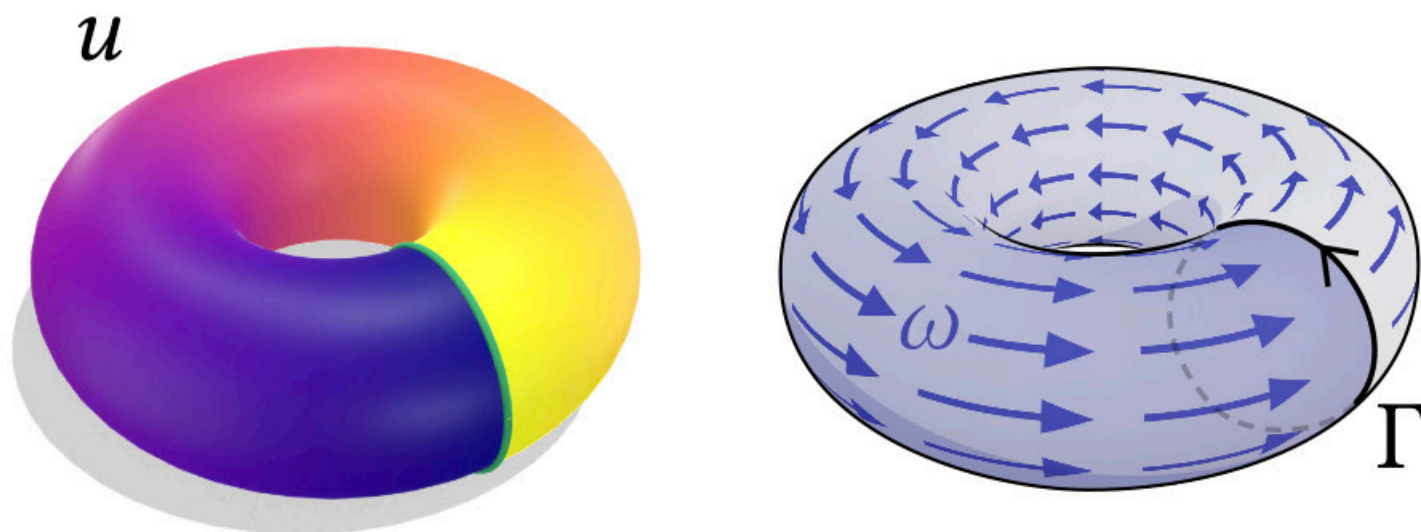
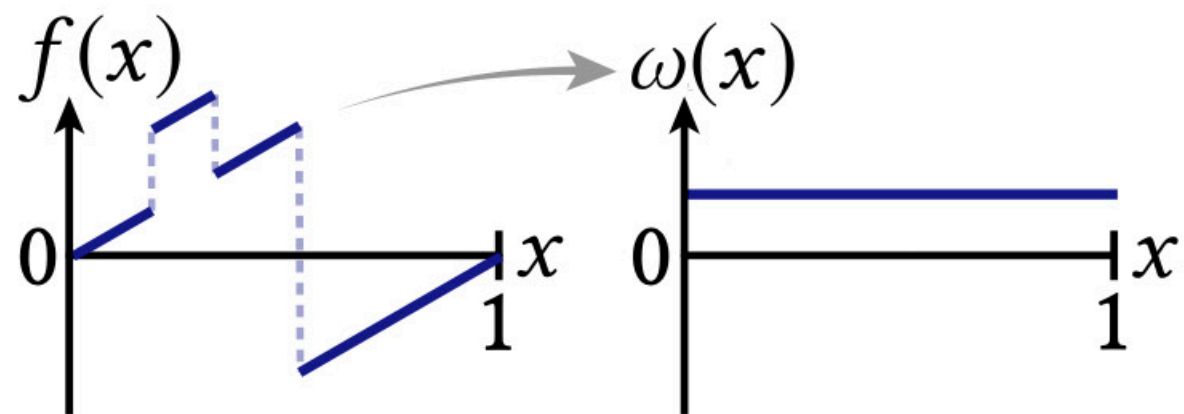
2D jump harmonic functions — same story

For a **jump harmonic** function f , the *Darboux derivative* $\omega := \mathcal{D}f$ "forgets" jumps:



2D jump harmonic functions – same story

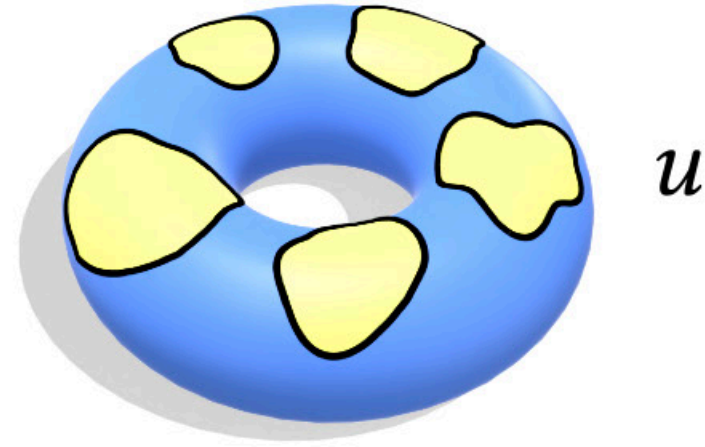
For a **jump harmonic function** f , the *Darboux derivative* $\omega := \mathcal{D}f$ "forgets" jumps:



Nonbounding curves \iff nonzero derivative

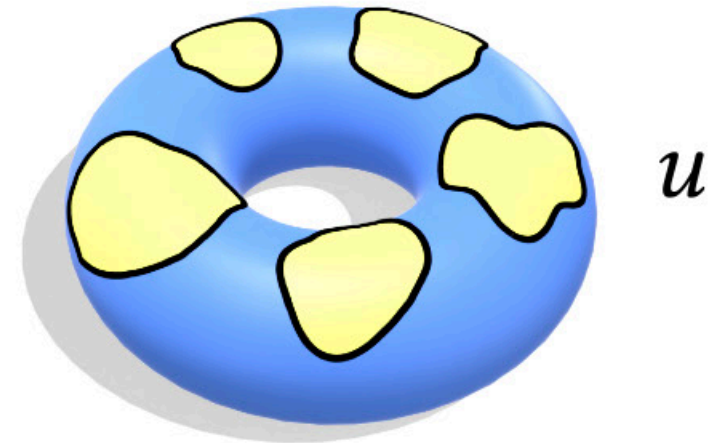
Nonbounding curves \iff nonzero derivative

If $\omega = 0$, then u is piecewise constant \implies
 u is already a valid region labeling.

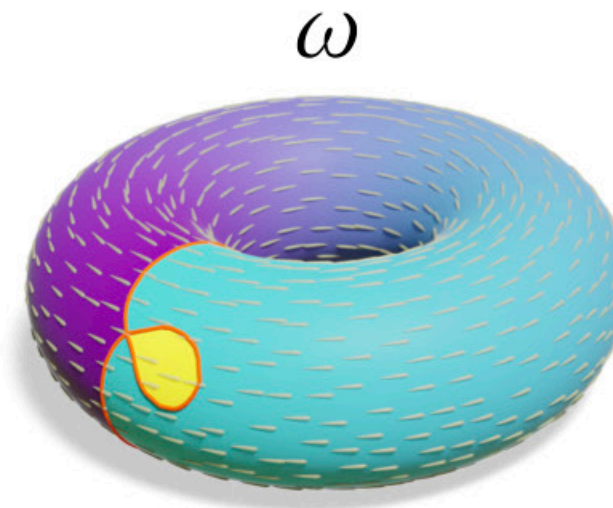
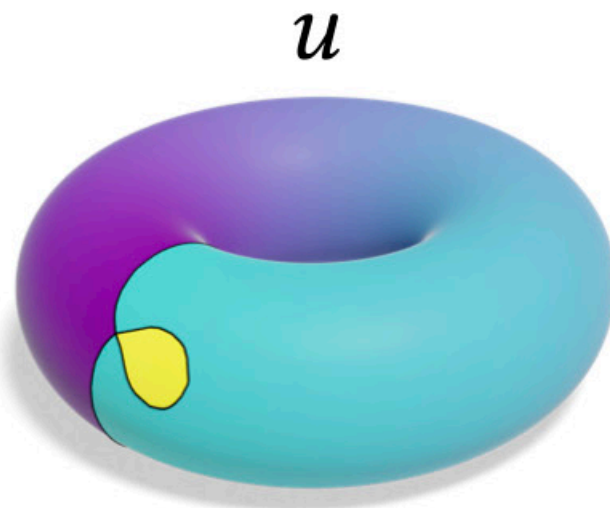


Nonbounding curves \iff nonzero derivative

If $\omega = 0$, then u is piecewise constant \implies
 u is already a valid region labeling.

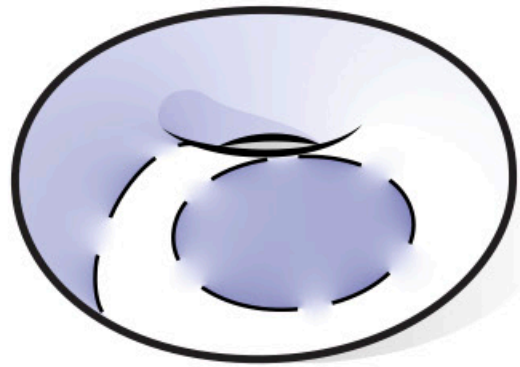


Otherwise, Γ has nonbounding components:

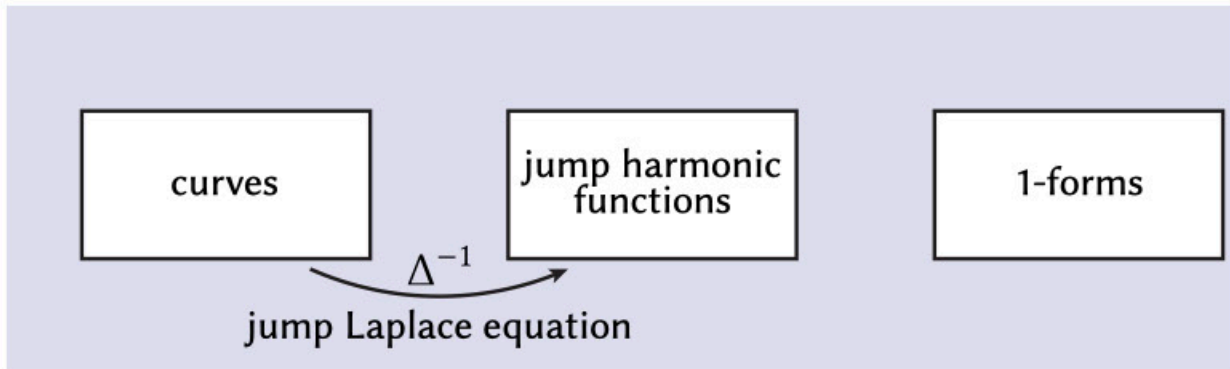


Derivative decomposition

Derivative decomposition

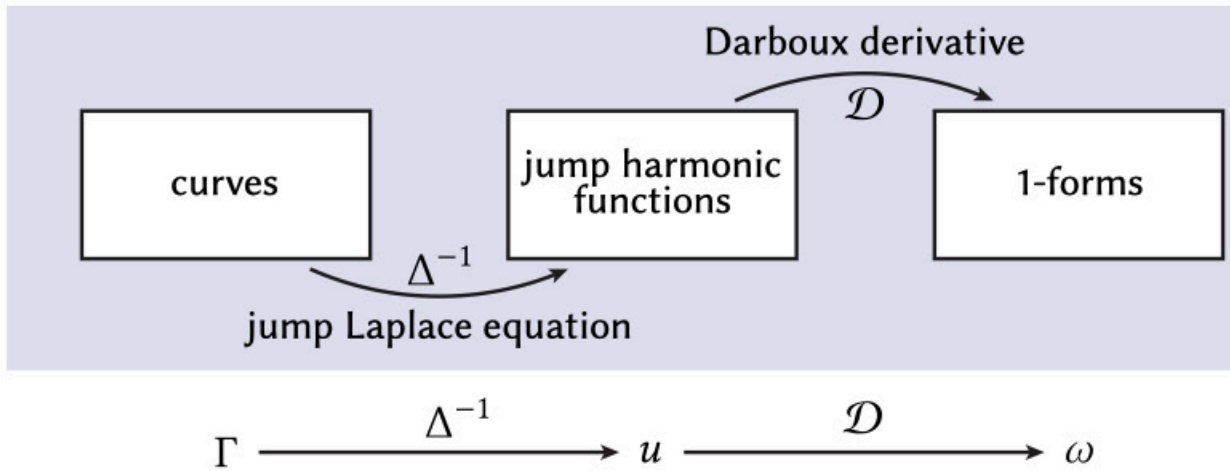
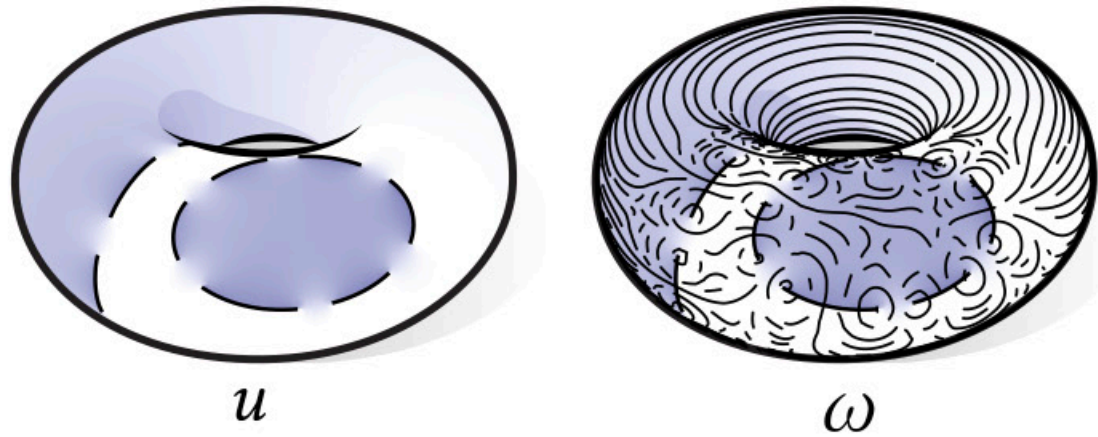


u

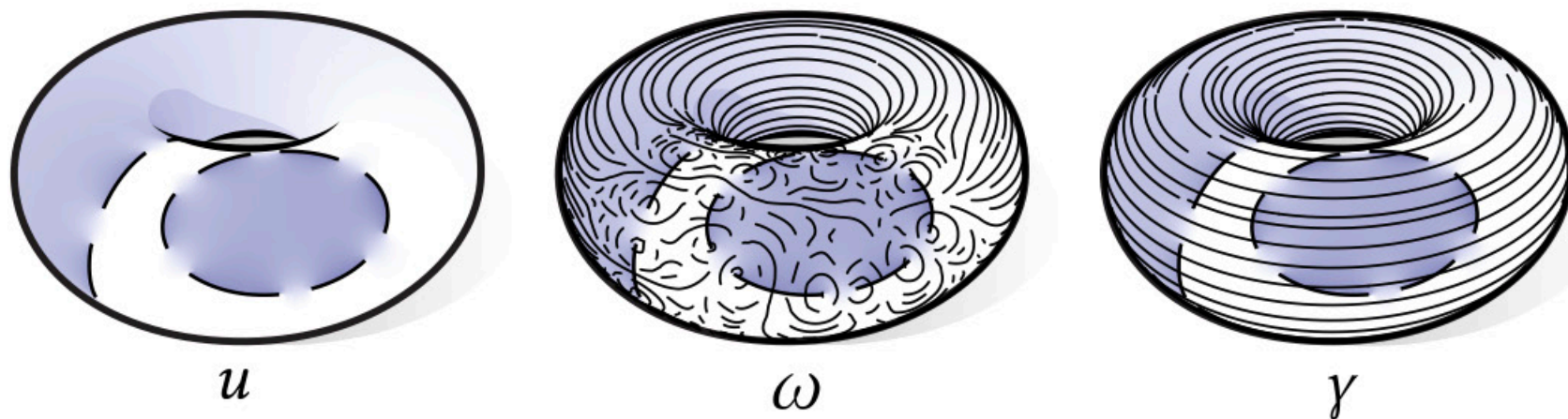


$$\Gamma \xrightarrow{\Delta^{-1}} u$$

Derivative decomposition



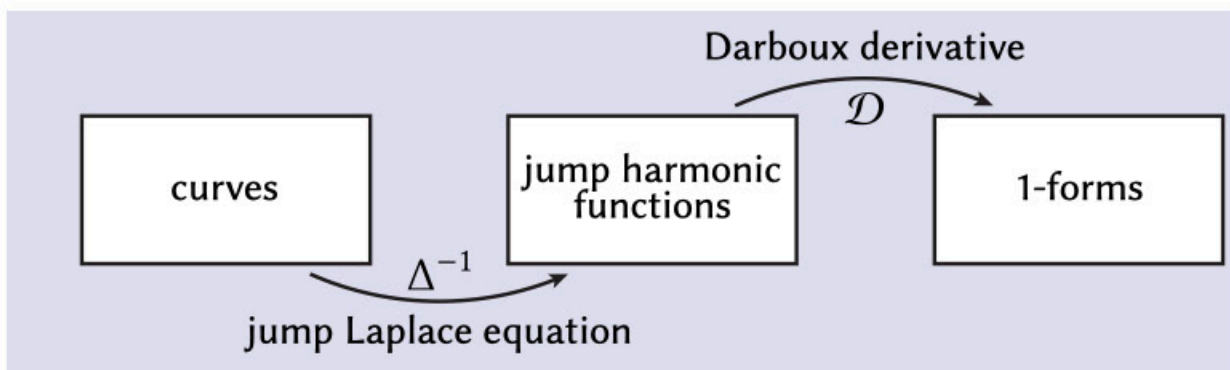
Derivative decomposition



Hodge decomposition:

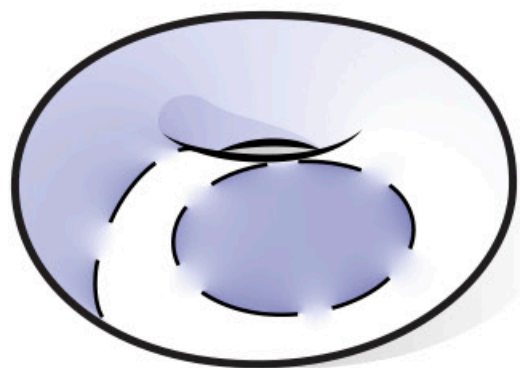
$$\omega = d\alpha + \delta\beta + \gamma$$

γ is a **harmonic 1-form**

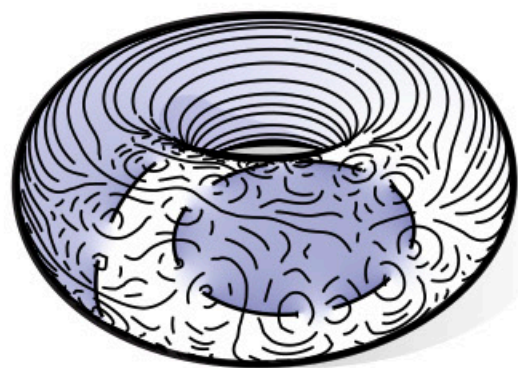


$$\Gamma \xrightarrow{\Delta^{-1}} u \xrightarrow{\mathcal{D}} \omega \downarrow \gamma$$

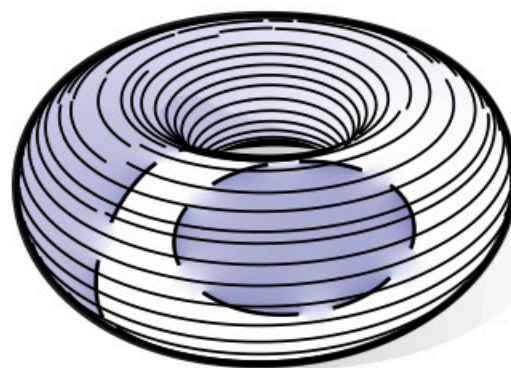
Derivative decomposition



u



ω

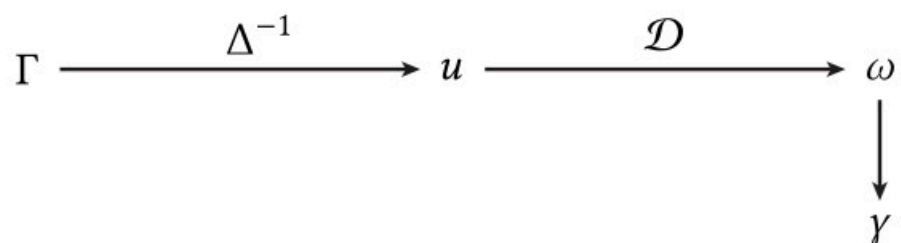
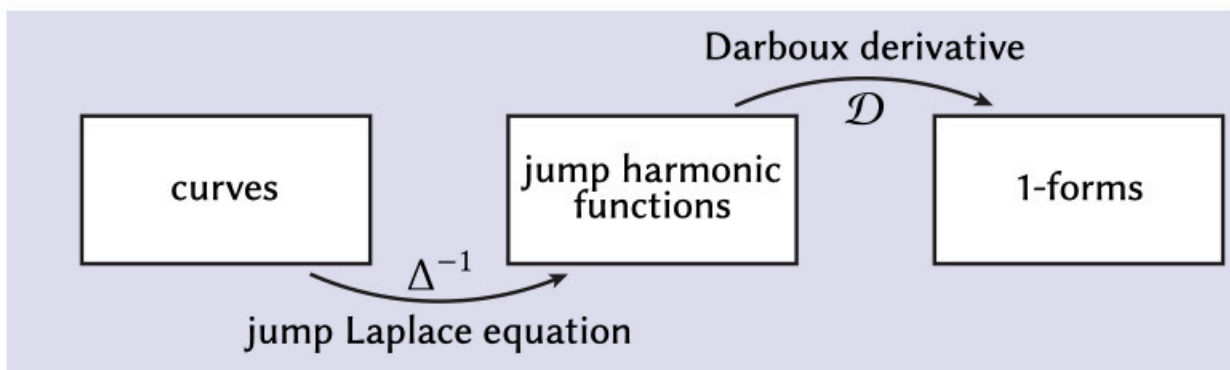


γ

Hodge decomposition:

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

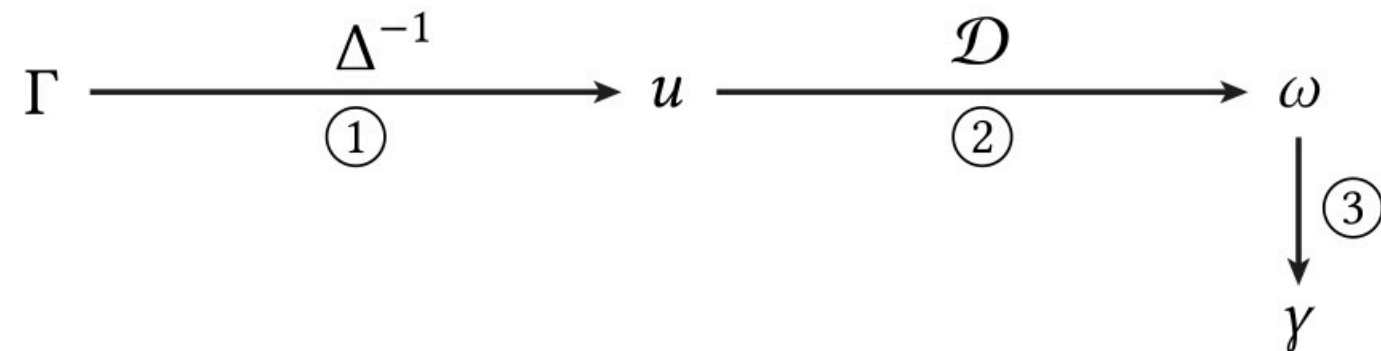
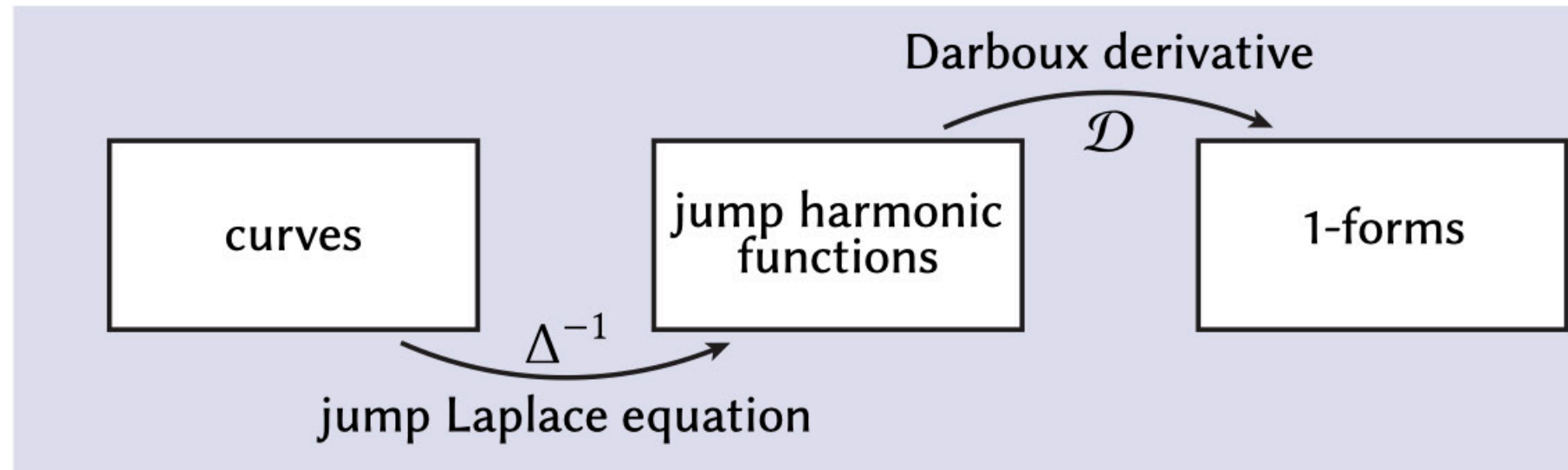
γ is a **harmonic 1-form**



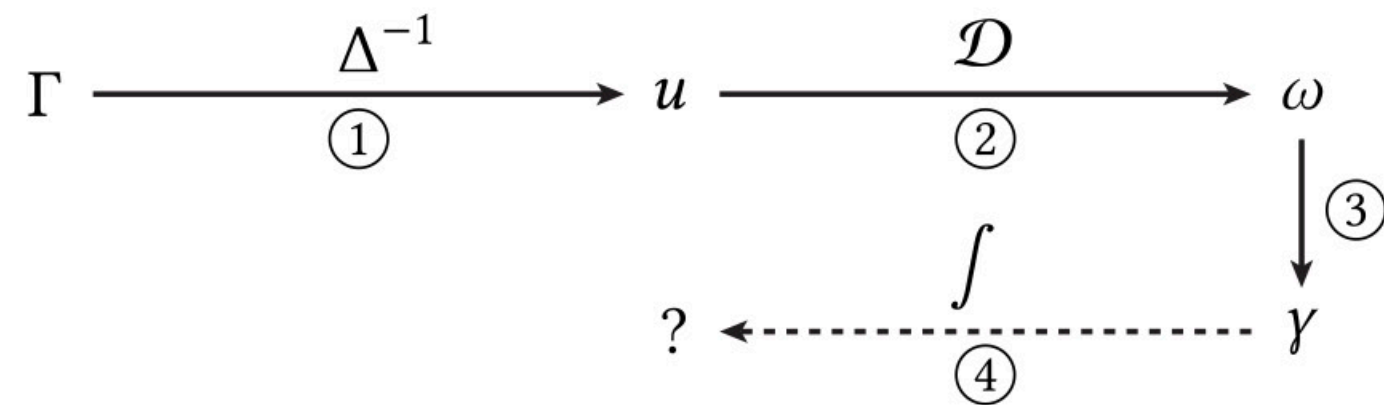
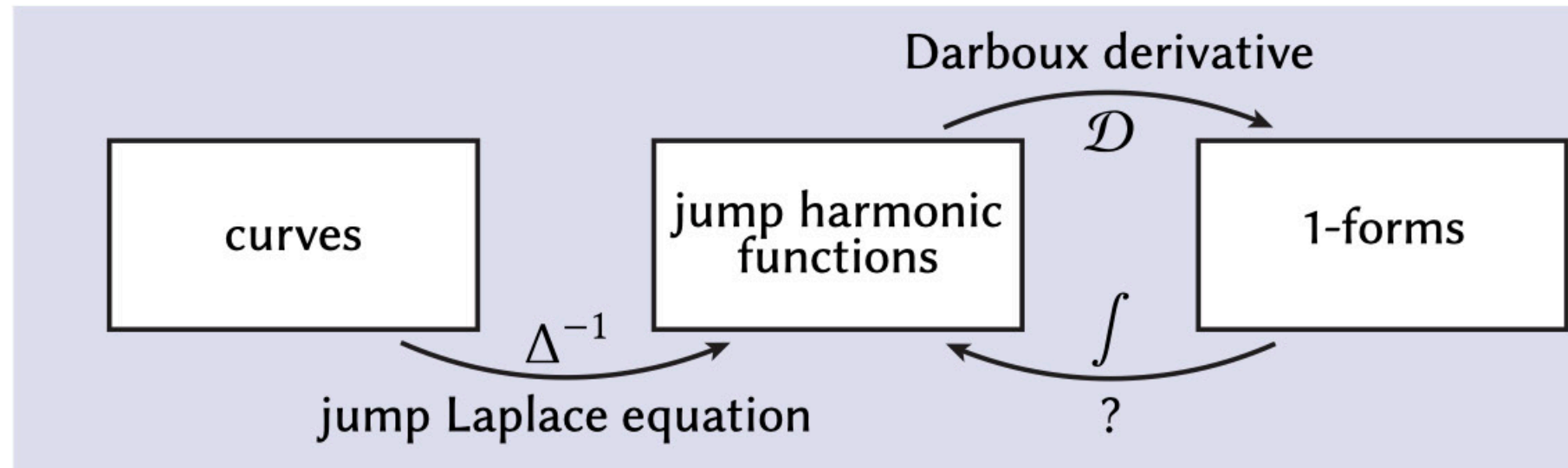
Nonbounding components of Γ are encoded by the harmonic component γ of ω .

More formally: (Non)bounding components of Γ correspond to 1-forms (non)congruent to zero in the *first cohomology group* $H^1(M) = \ker(d_1) \setminus \text{im}(d_0)$.

Derivative decomposition \rightarrow curve decomposition



Derivative decomposition \rightarrow curve decomposition



1-forms \rightarrow jump harmonic functions

Search for a scalar potential v that could have generated γ .

$$\mathcal{D}v = \gamma$$

1-forms \rightarrow jump harmonic functions

Search for a scalar potential v that could have generated γ .

$$\mathcal{D}v = \gamma$$

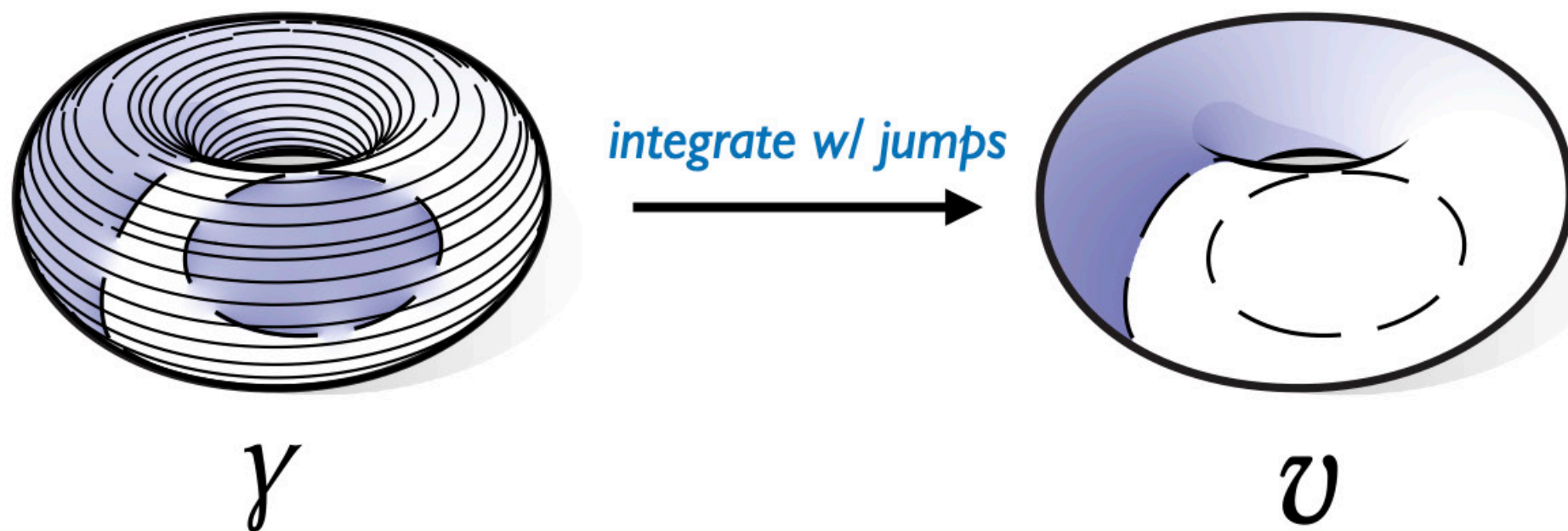
Since γ is harmonic, v must jump somewhere.

1-forms \rightarrow jump harmonic functions

Search for a scalar potential v that could have generated γ .

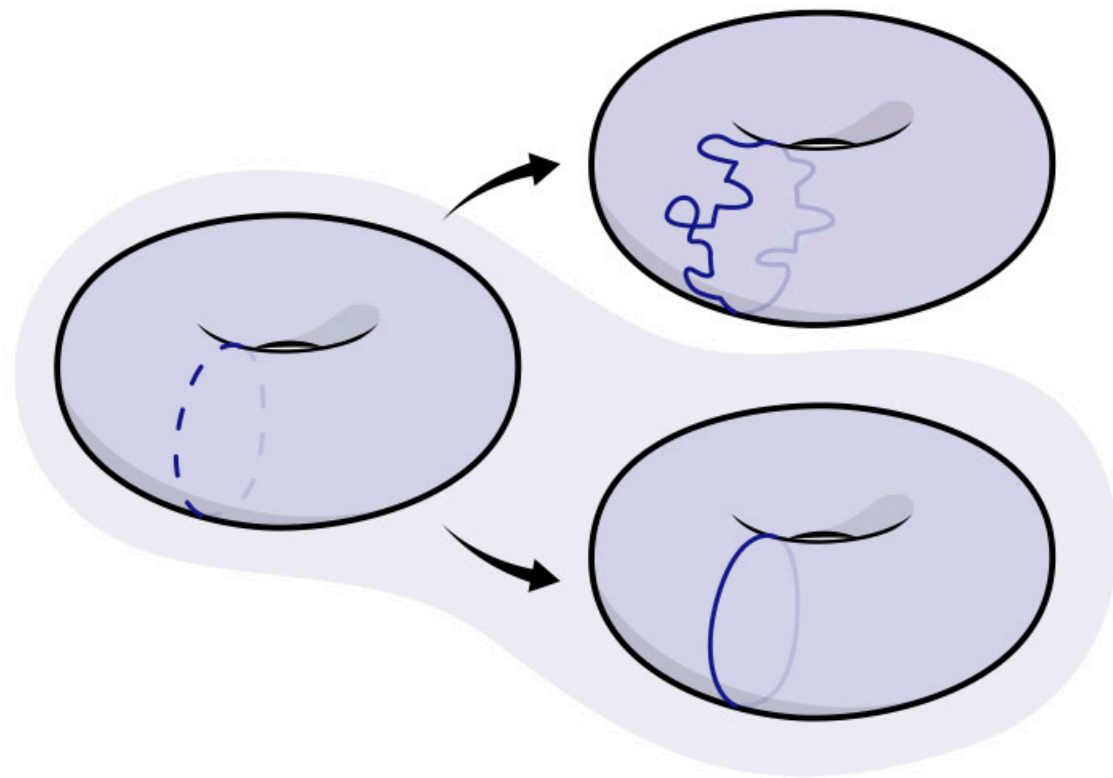
$$Dv = \gamma$$

Since γ is harmonic, v must jump somewhere.

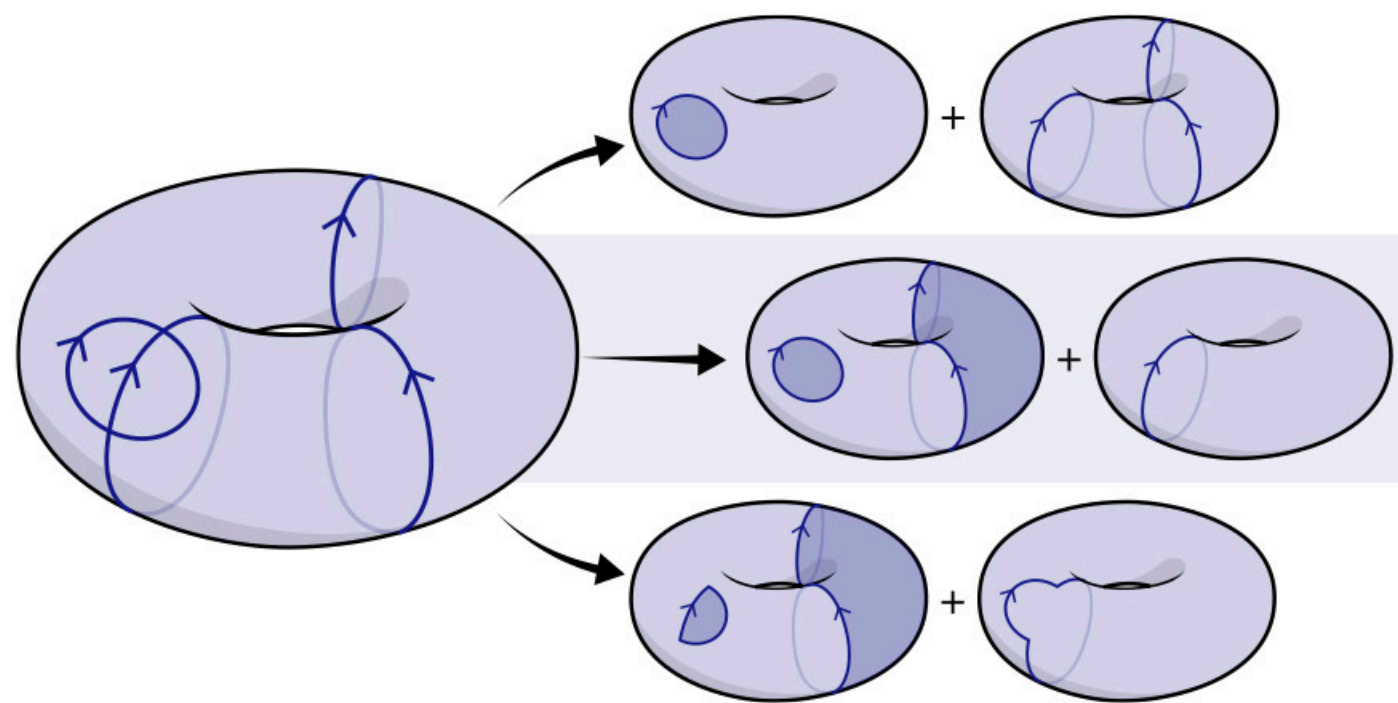
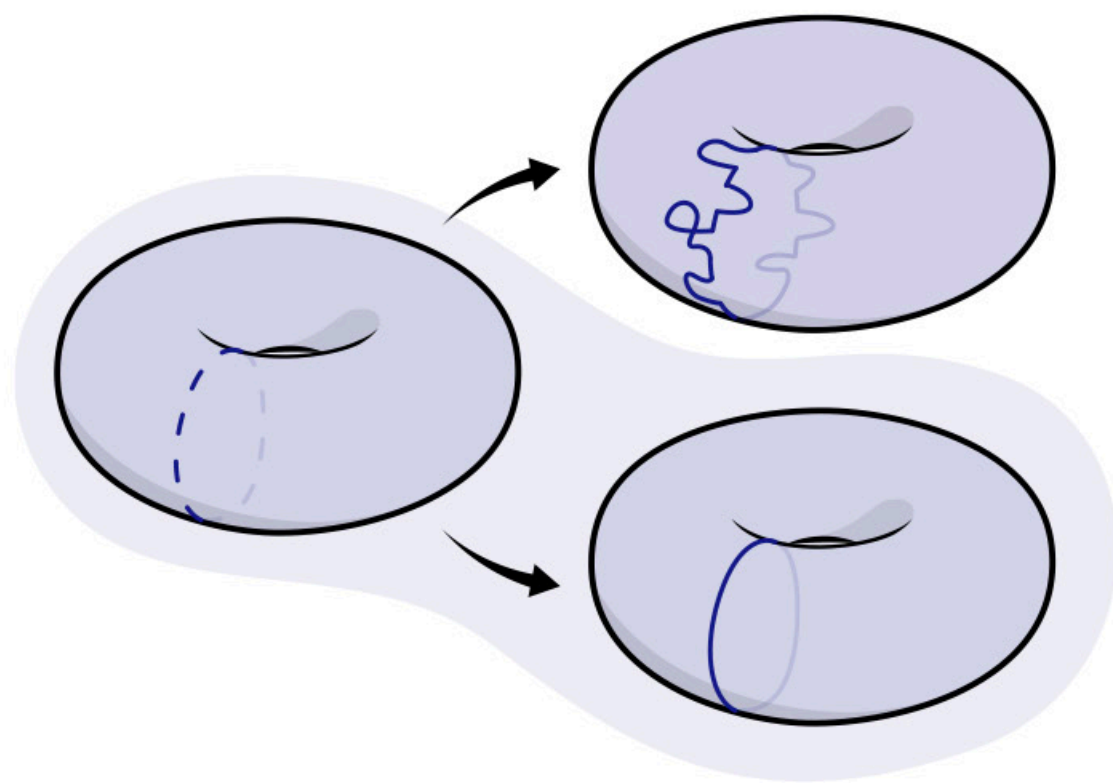


1-forms \rightarrow jump harmonic functions

1-forms \rightarrow jump harmonic functions

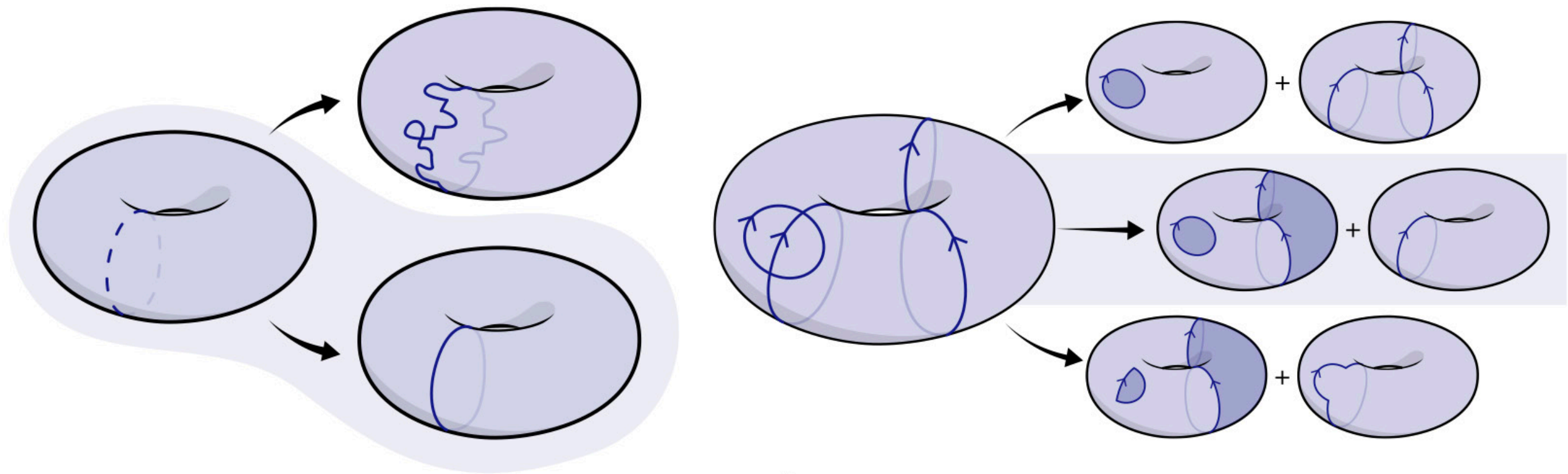


1-forms \rightarrow jump harmonic functions



1-forms \rightarrow jump harmonic functions

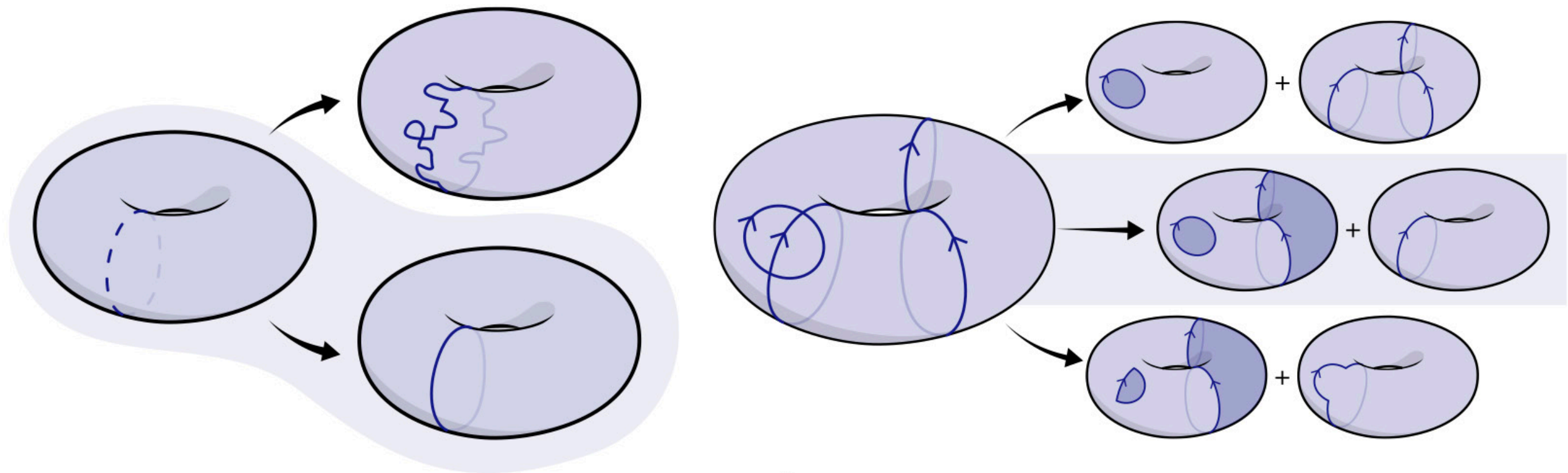
$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$



1-forms \rightarrow jump harmonic functions

penalize jumps

$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$

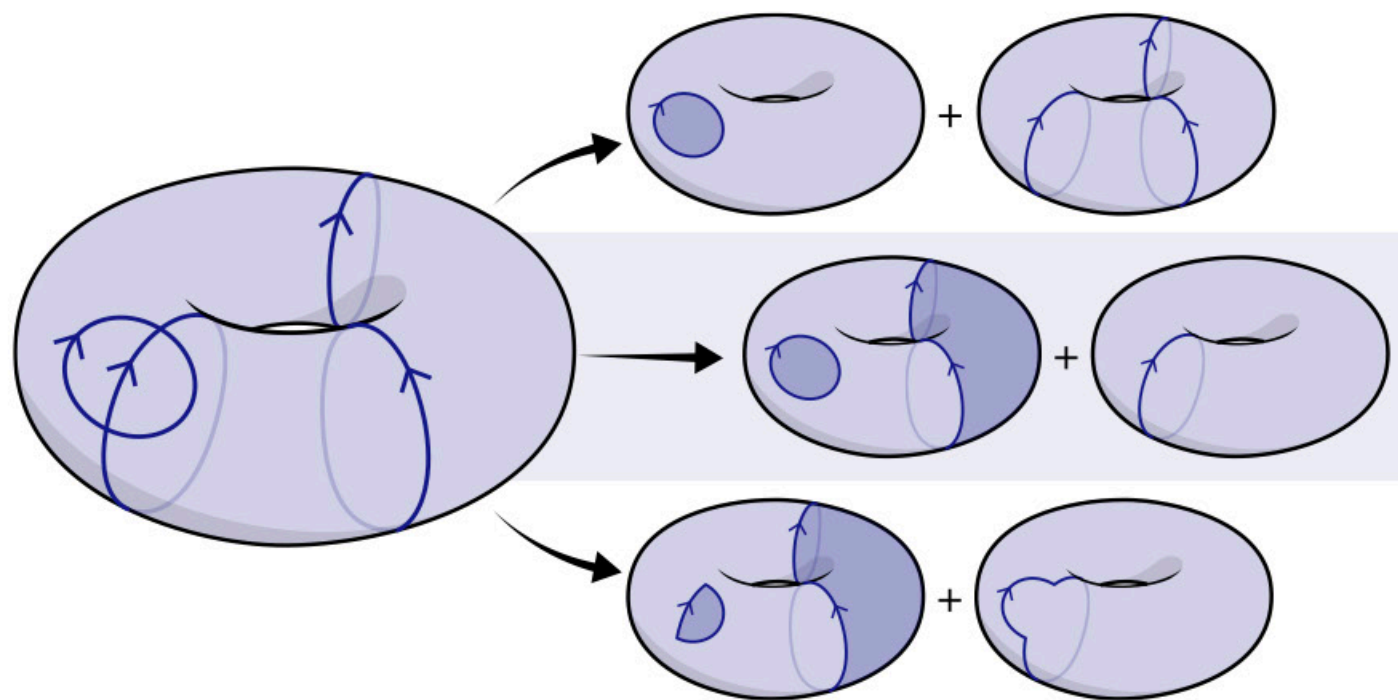
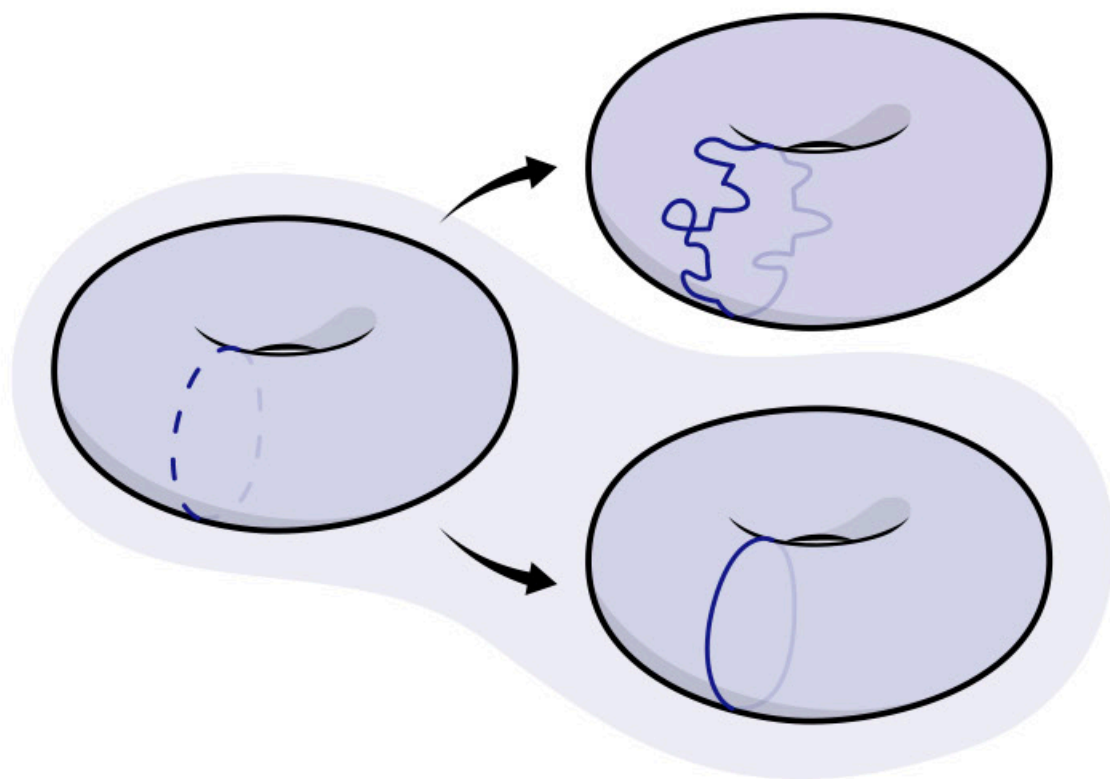


1-forms \rightarrow jump harmonic functions

penalize jumps

smaller penalty across Γ

$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$



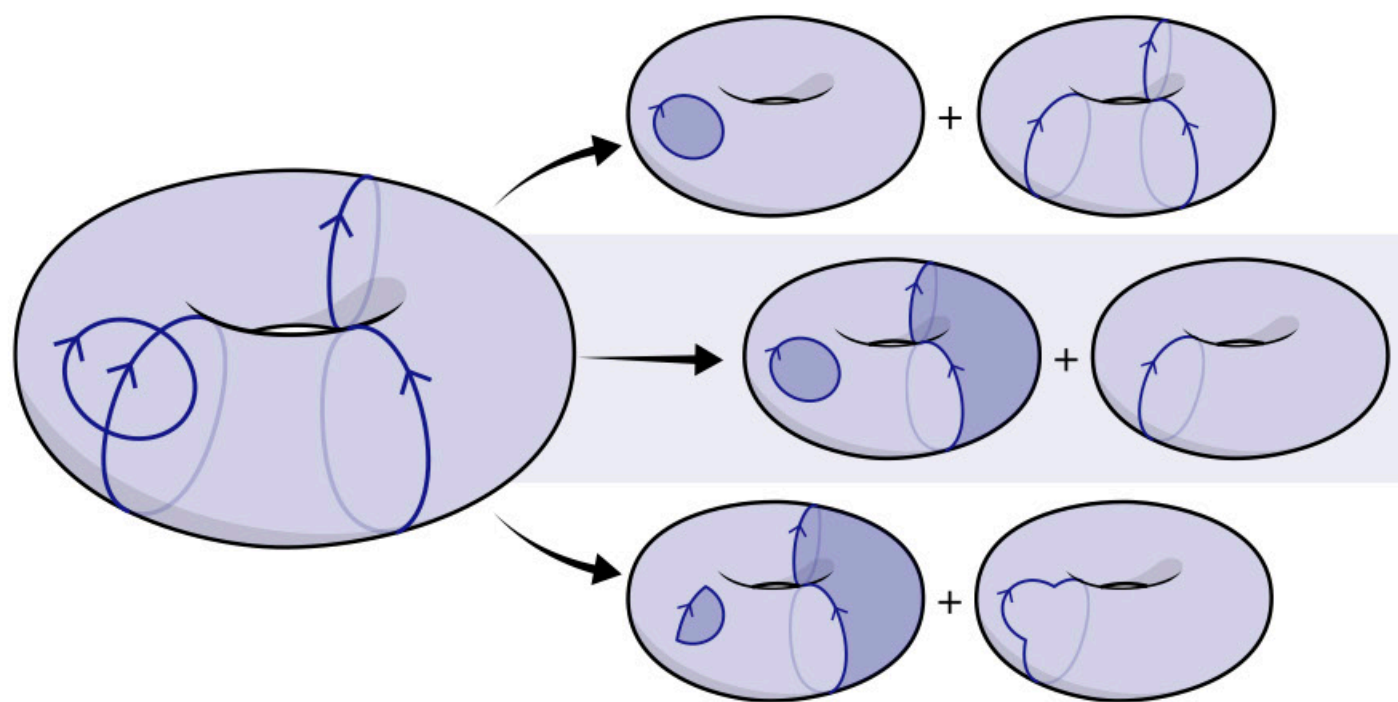
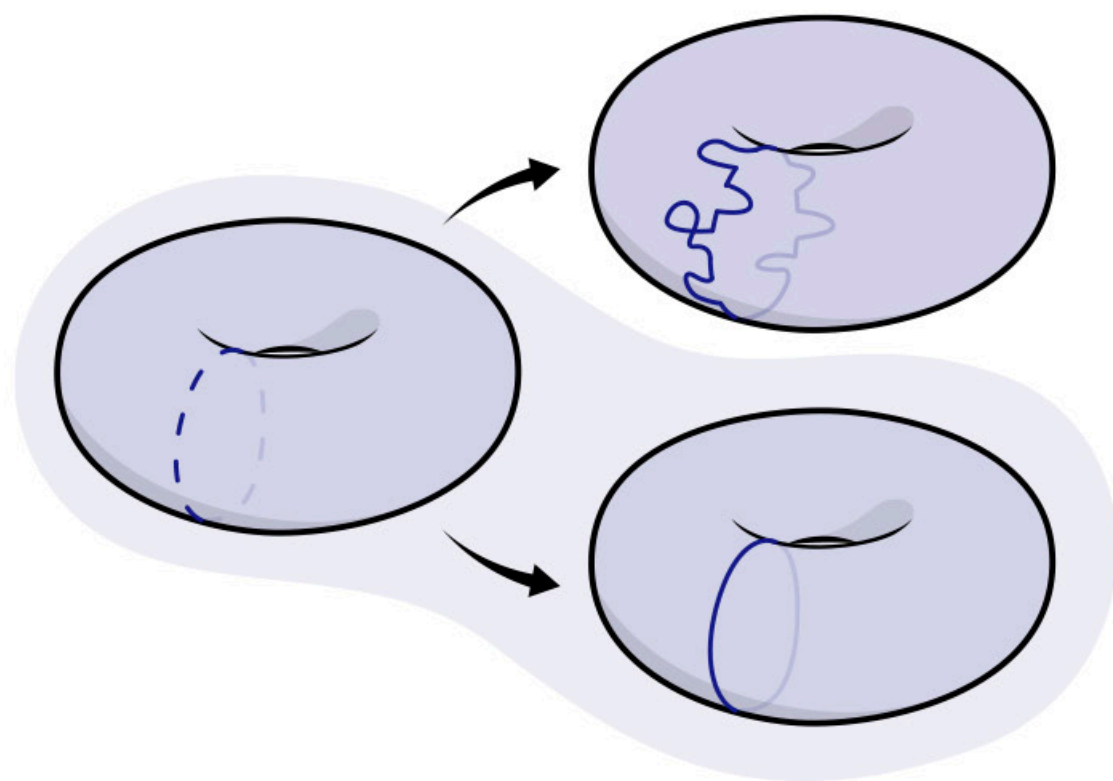
1-forms \rightarrow jump harmonic functions

penalize jumps

smaller penalty across Γ

$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$

concentrate jumps across Γ



1-forms \rightarrow jump harmonic functions

penalize jumps

smaller penalty across Γ

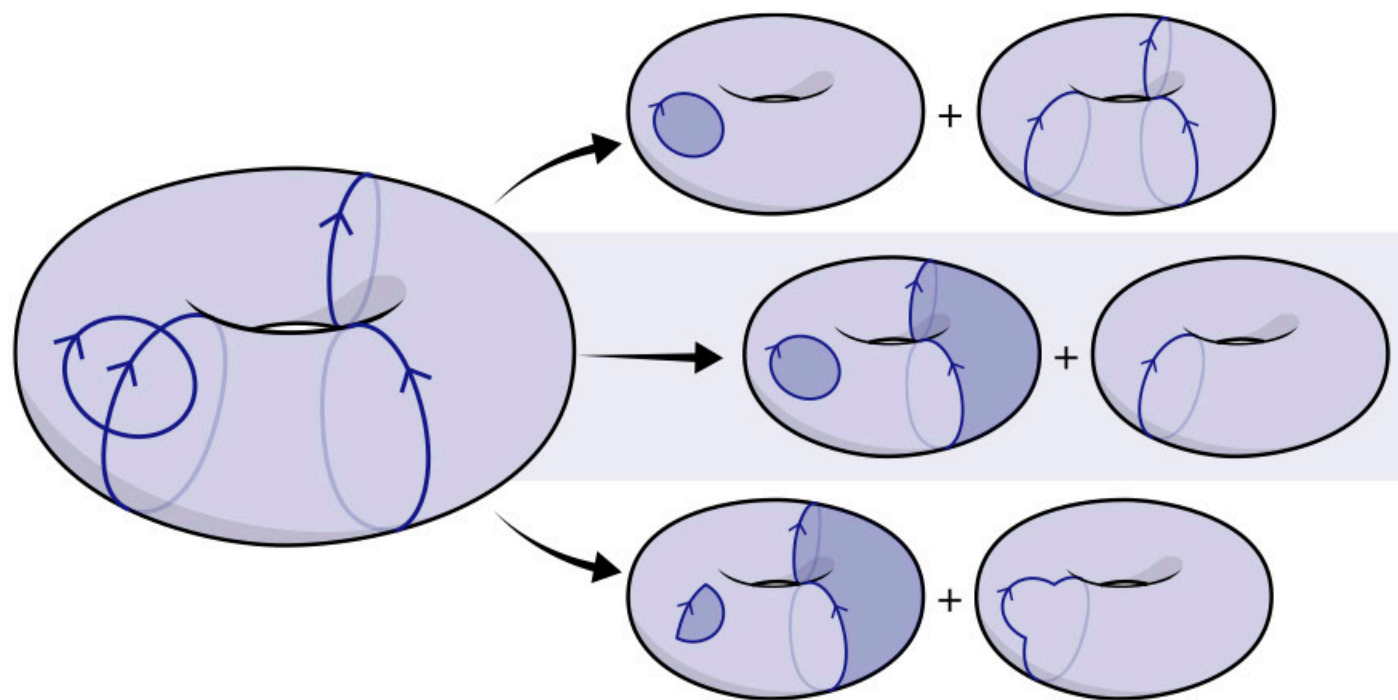
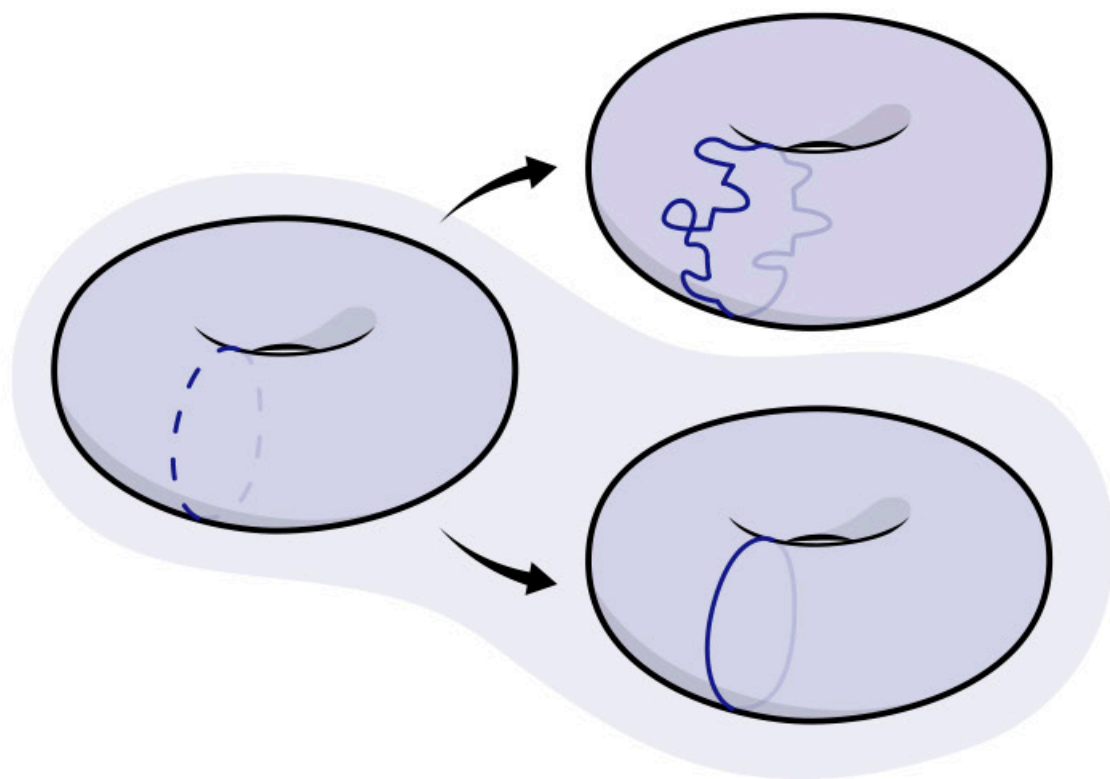
$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$

concentrate jumps across Γ

subject to

$$Dv = \gamma$$

(co)homology constraint



1-forms \rightarrow jump harmonic functions

pick shortest completion in homology class

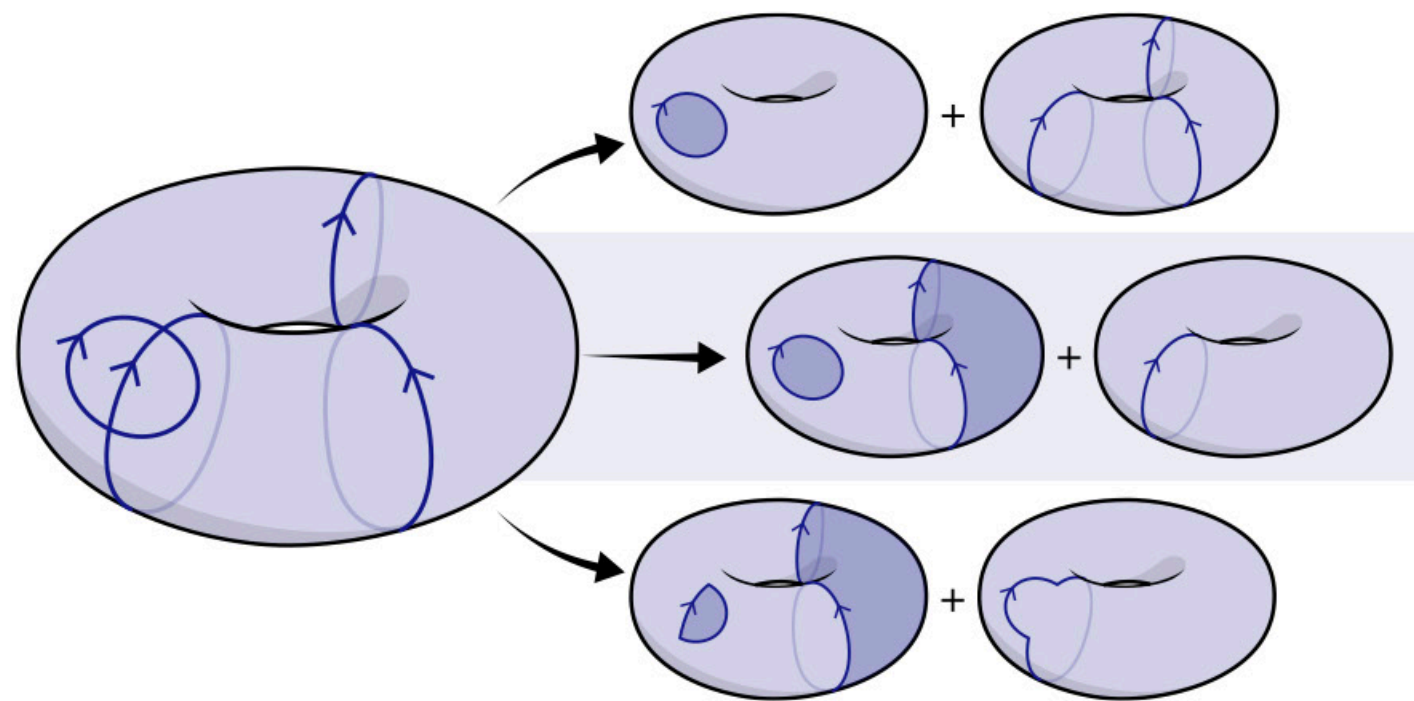
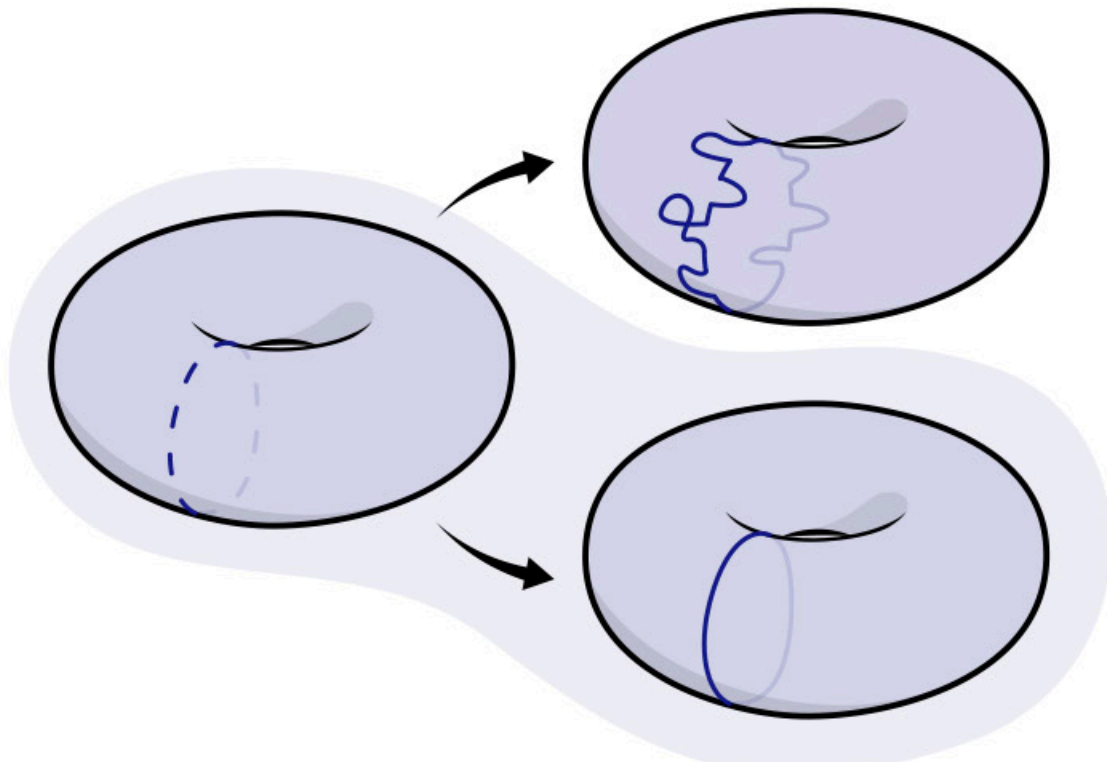
$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$

penalize jumps *smaller penalty across Γ*

concentrate jumps across Γ

subject to $Dv = \gamma$

(co)homology constraint



1-forms \rightarrow jump harmonic functions

pick shortest completion in homology class

$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$

penalize jumps

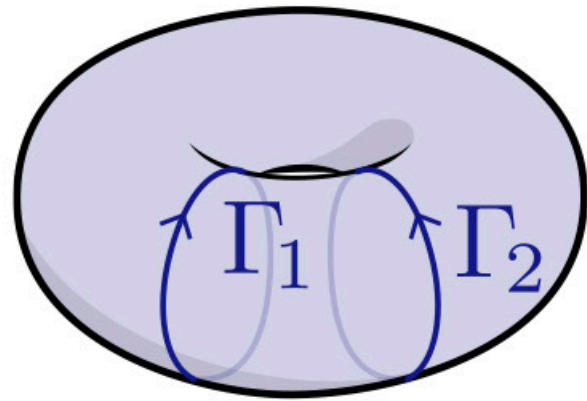
smaller penalty across Γ

concentrate jumps across Γ

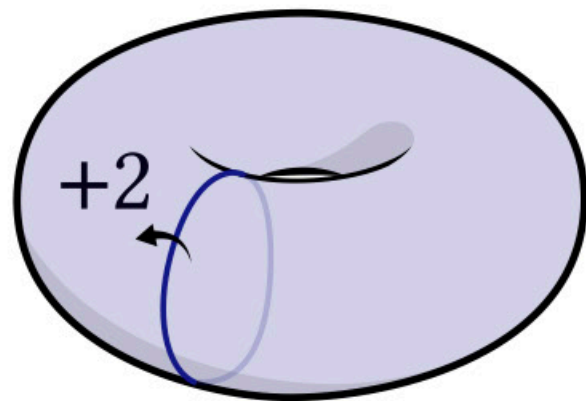
subject to

$$Dv = \gamma$$

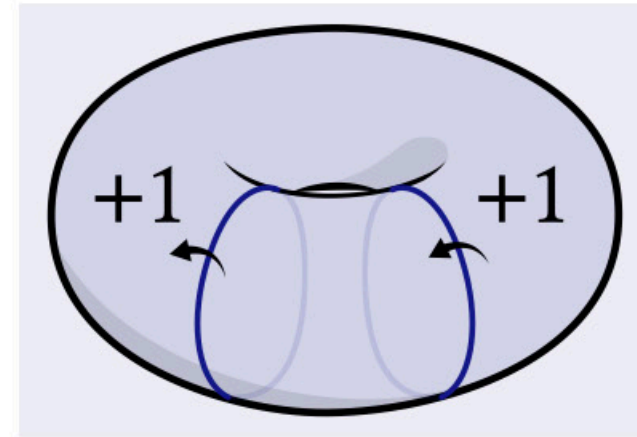
(co)homology constraint



input



v



1-forms \rightarrow jump harmonic functions

pick shortest completion in homology class

$$\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$$

penalize jumps

smaller penalty across Γ

concentrate jumps across Γ

subject to

$$\mathcal{D}v = \gamma$$

(co)homology constraint

$$0 \leq \frac{v^+ - v^-}{u^+ - u^-} \leq 1 \quad \text{on } \Gamma \quad \text{no extra loops}$$

1-forms \rightarrow jump harmonic functions

pick shortest completion in homology class

$\min_{v: M \rightarrow \mathbb{R}}$

penalize jumps

$$\int |\text{the jumps not across } \Gamma|$$

$$+ \varepsilon \int |\text{the jumps across } \Gamma|$$

smaller penalty across Γ

concentrate jumps across Γ

subject to

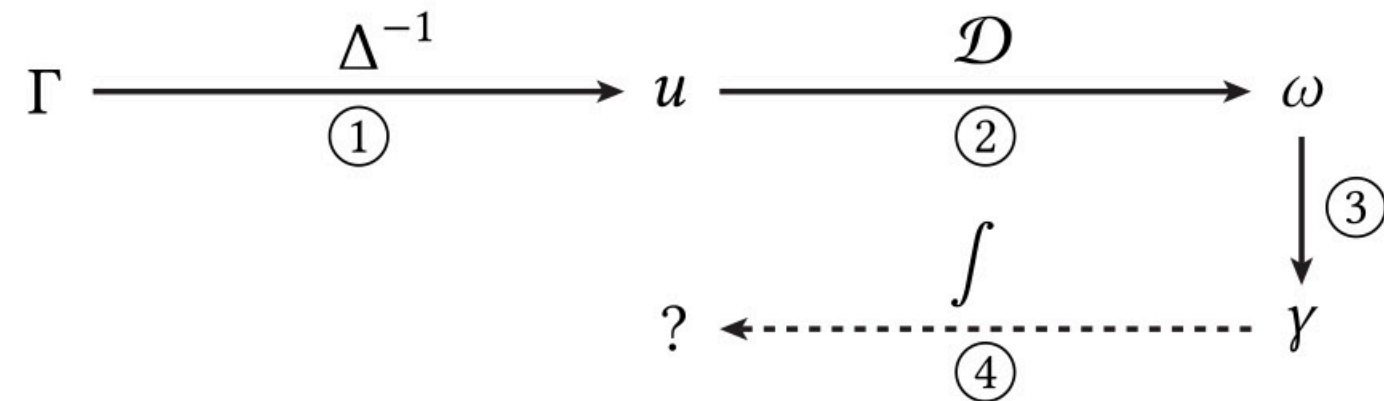
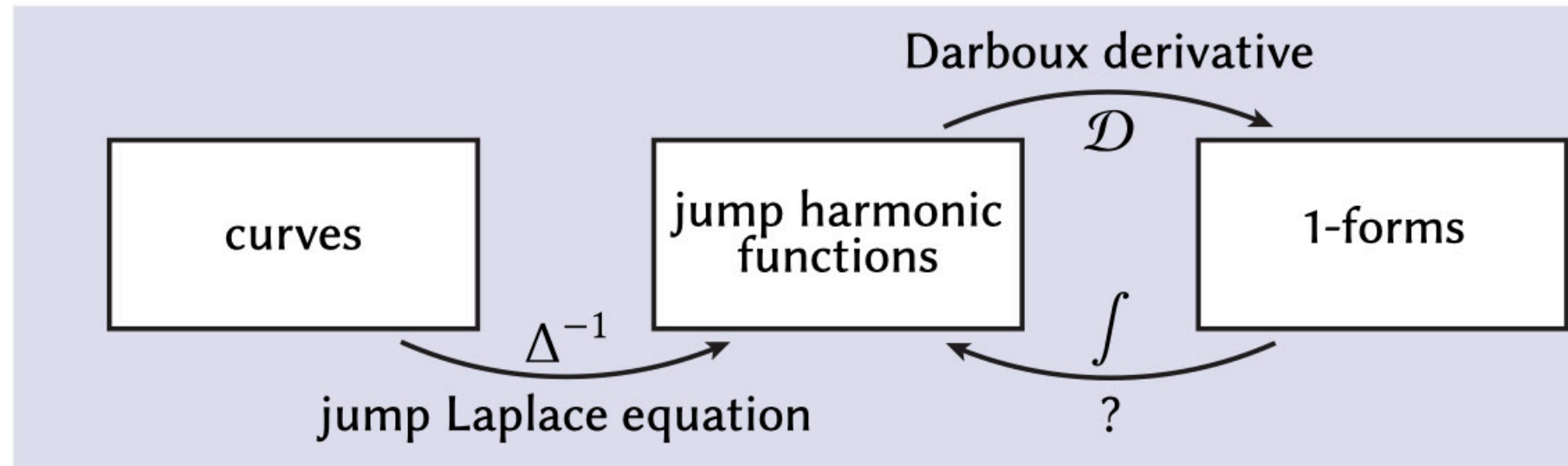
$$\mathcal{D}v = \gamma$$

(co)homology constraint

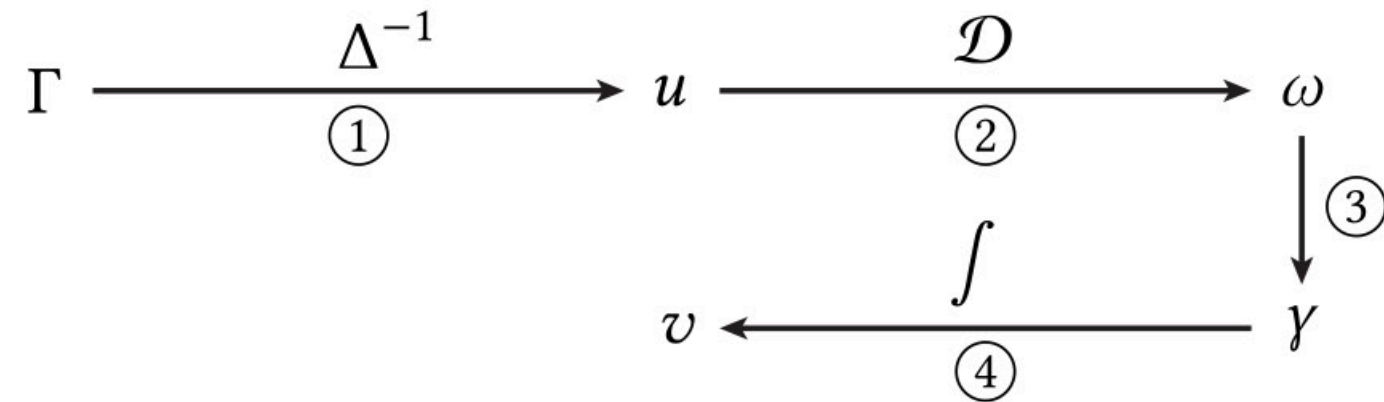
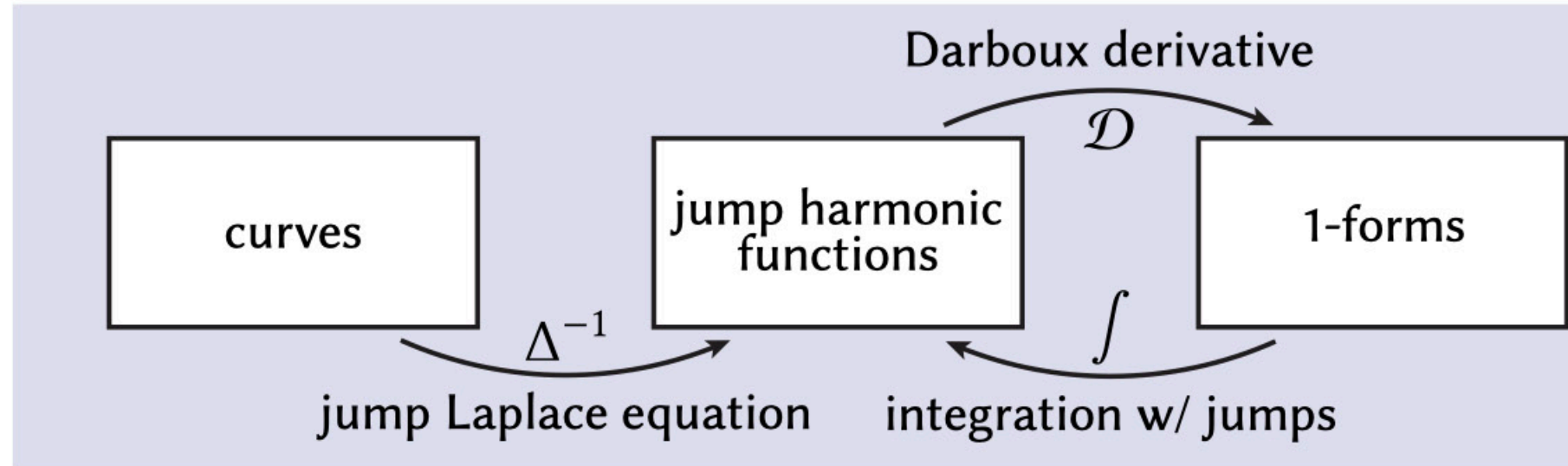
$$0 \leq \frac{v^+ - v^-}{u^+ - u^-} \leq 1 \quad \text{on } \Gamma \quad \textit{no extra loops}$$

"residual function"

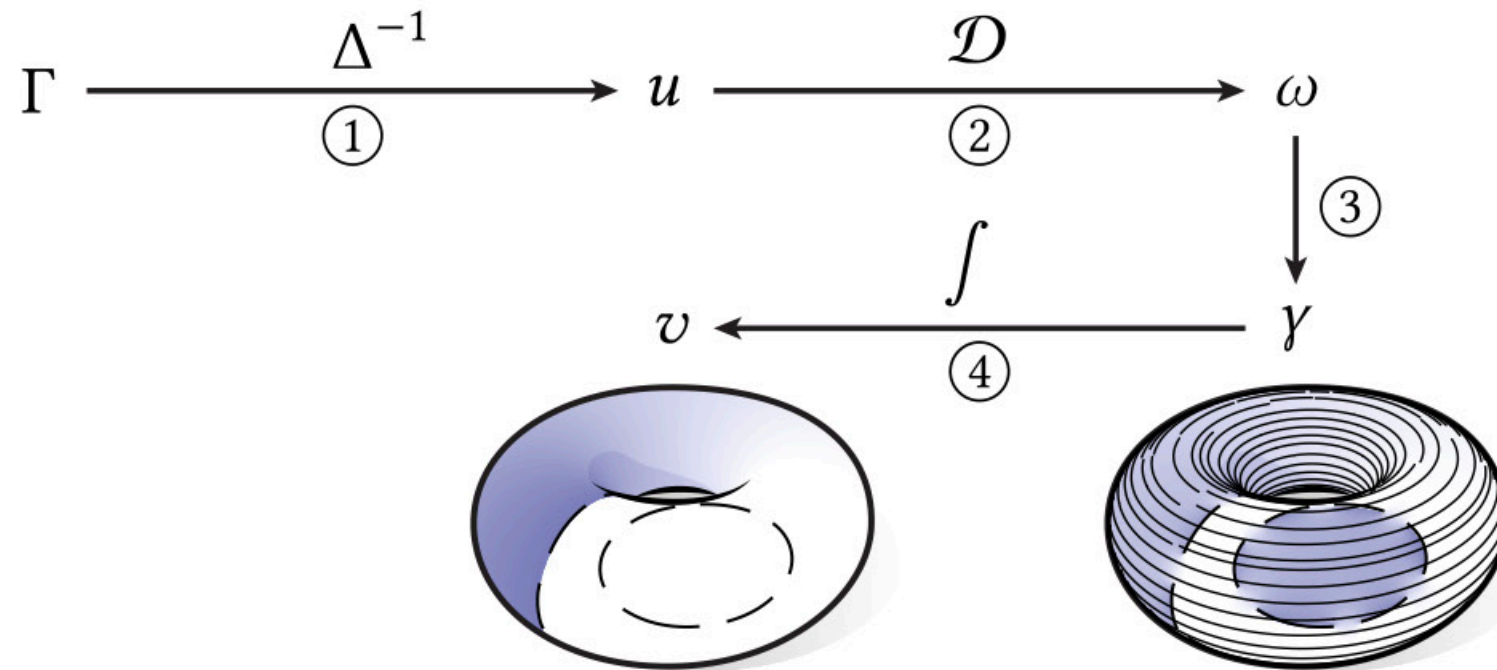
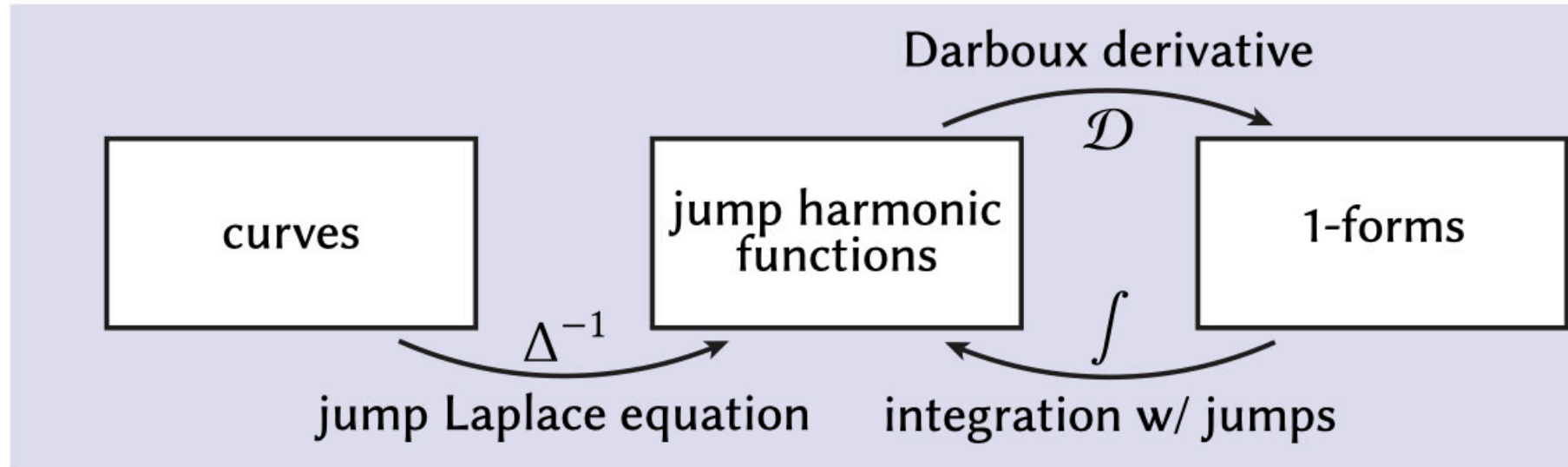
1-forms \rightarrow jump harmonic functions



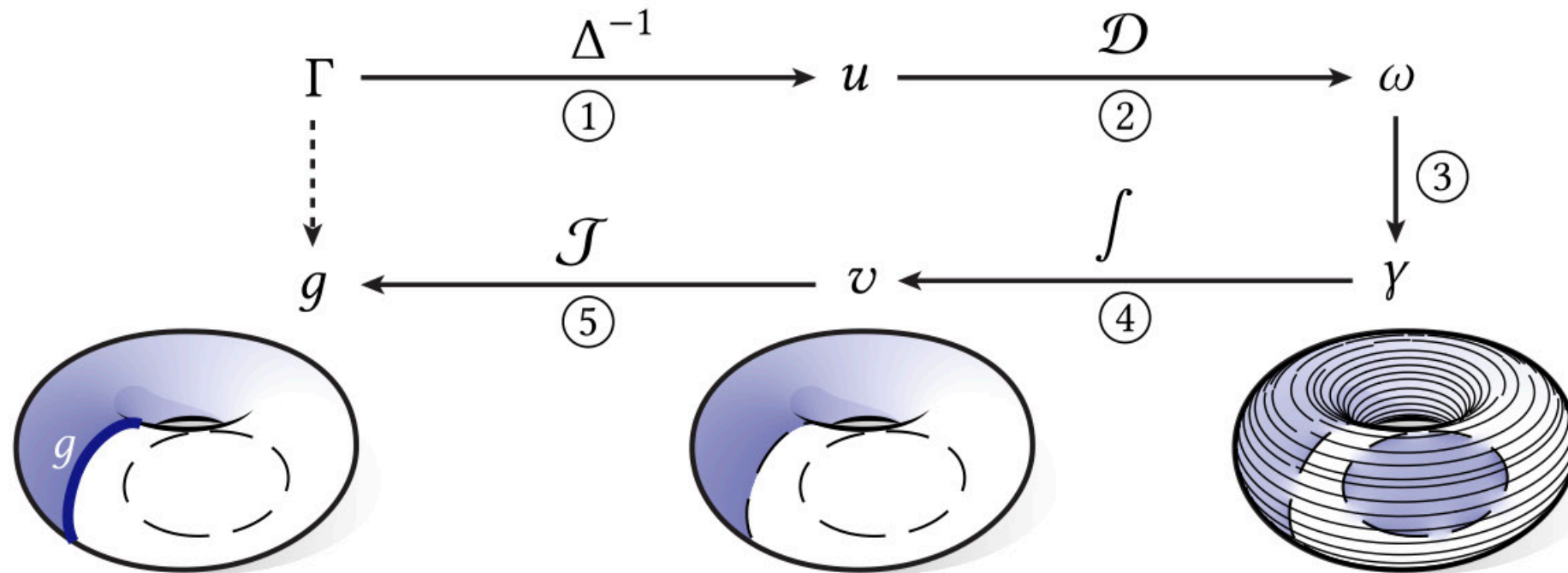
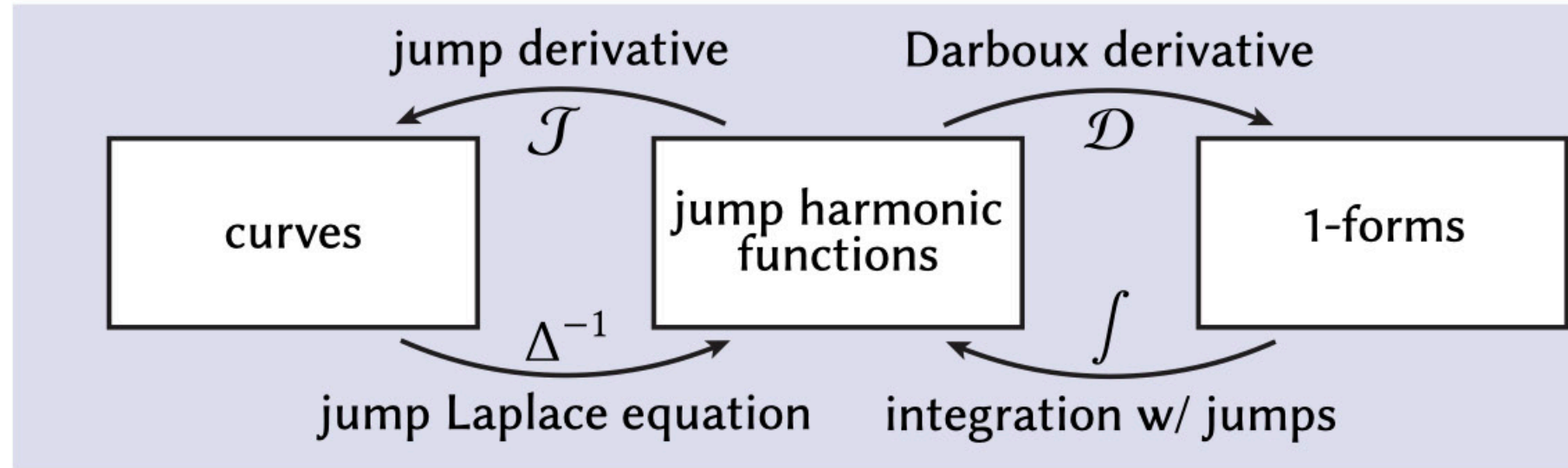
1-forms \rightarrow jump harmonic functions



1-forms \rightarrow jump harmonic functions



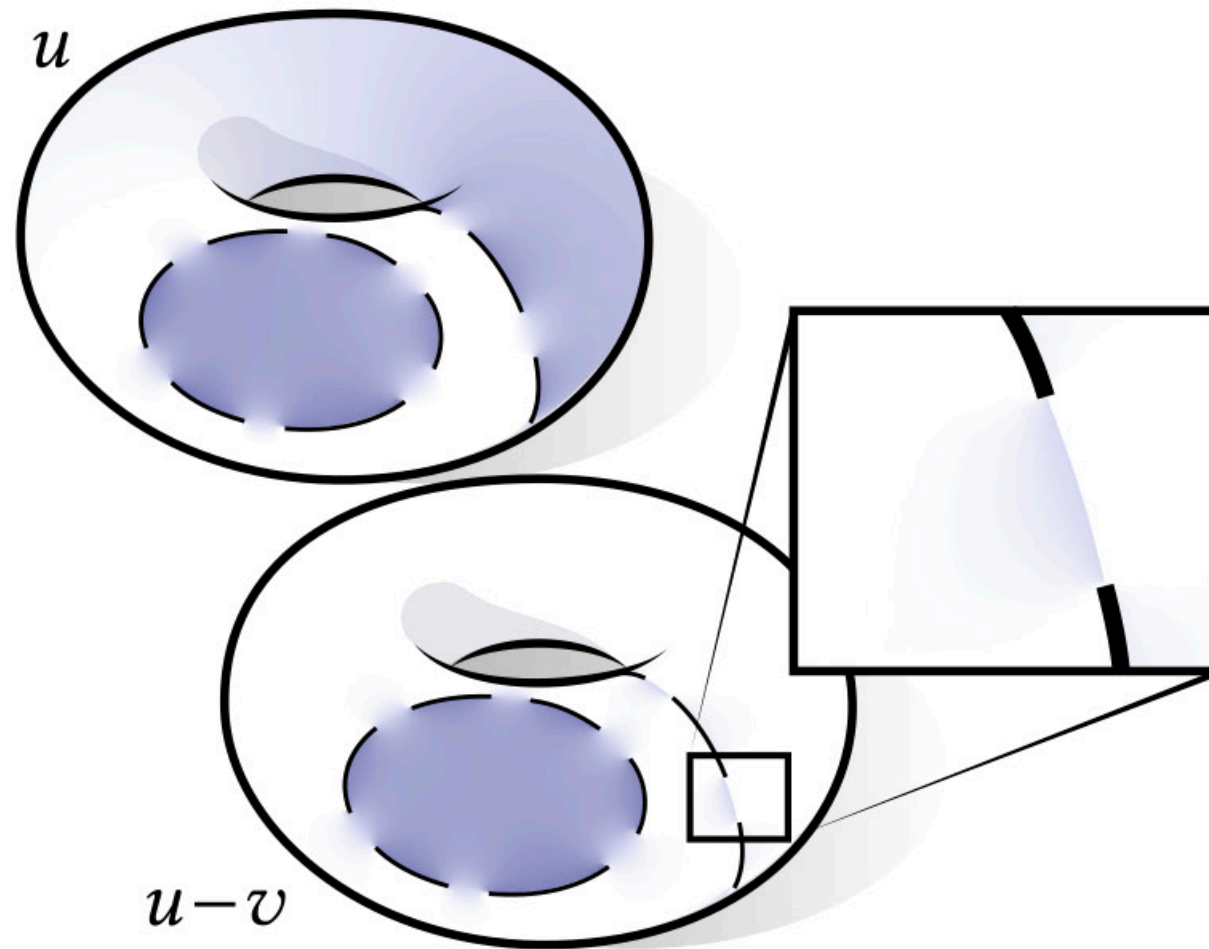
Jump harmonic function \rightarrow curve decomposition



Winding number function

Winding number function

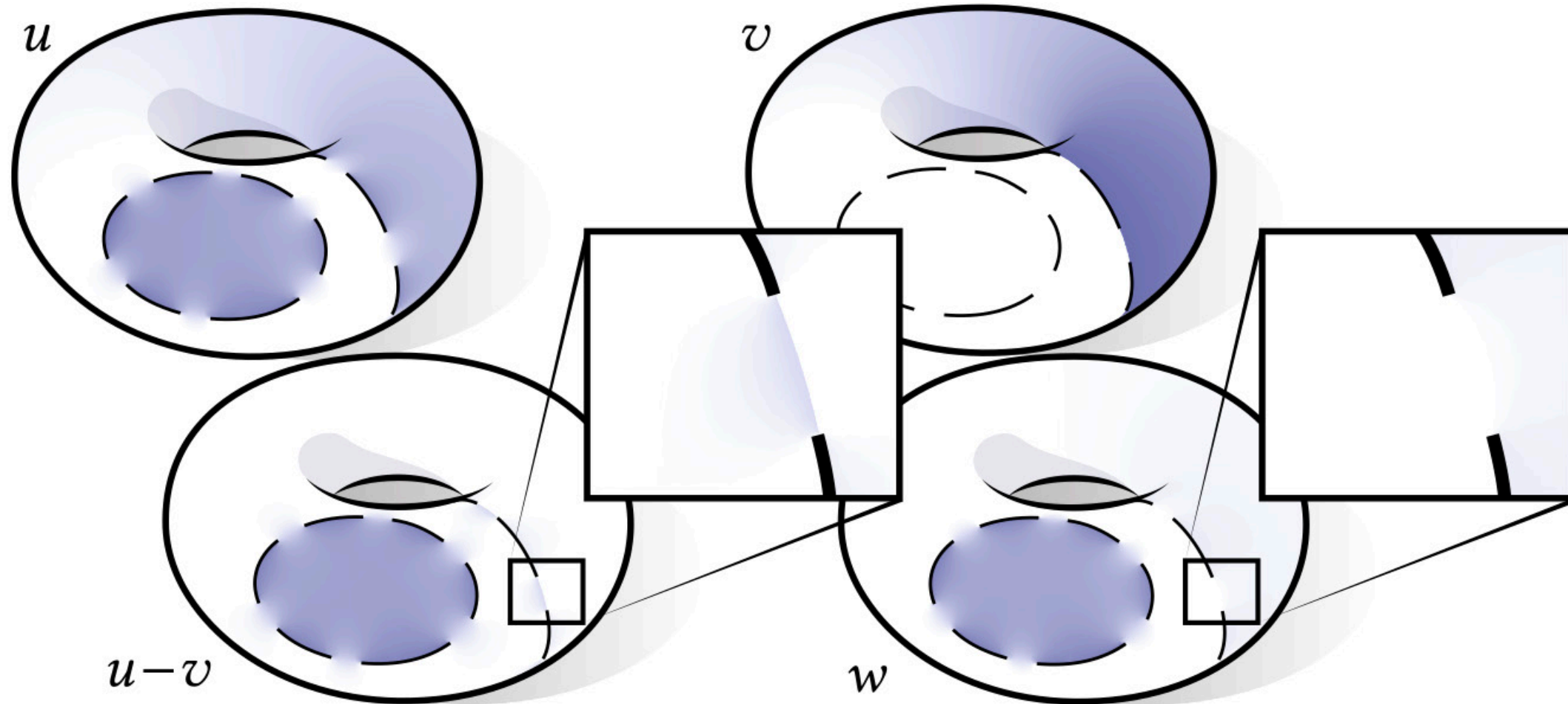
Simply subtracting the residual function yields extraneous discontinuities.



Winding number function

Simply subtracting the residual function yields extraneous discontinuities.

Solution: Solve for a new harmonic function w with jumps only across Γ .



Completing the round-trip

curves

jump harmonic
functions

1-forms

Completing the round-trip

differentiate

curves

jump harmonic
functions

1-forms

integrate

Completing the round-trip

differentiate

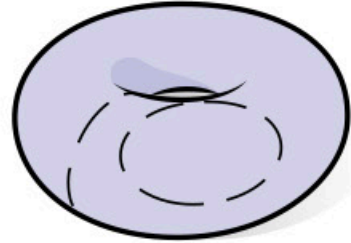
curves

jump harmonic
functions

1-forms

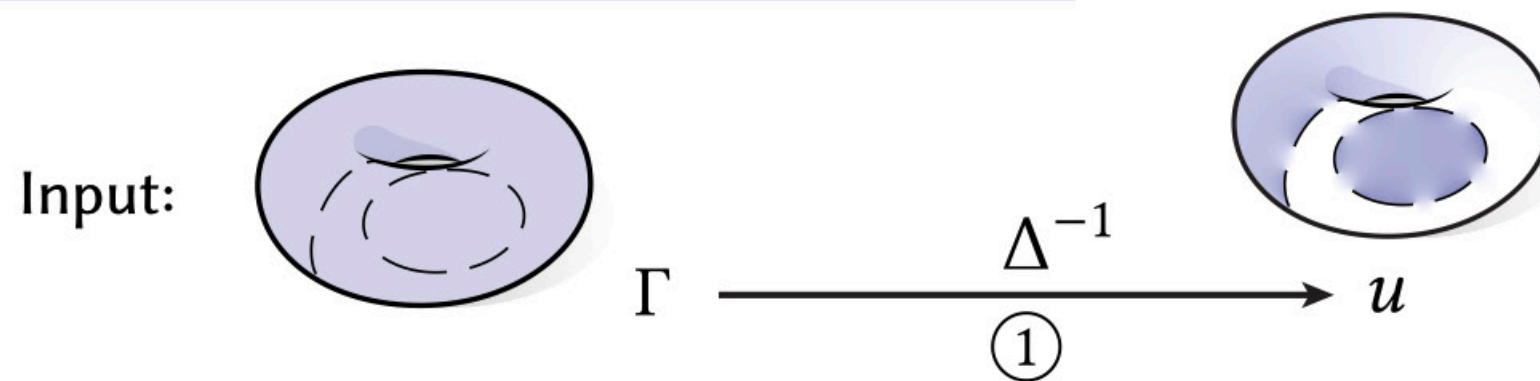
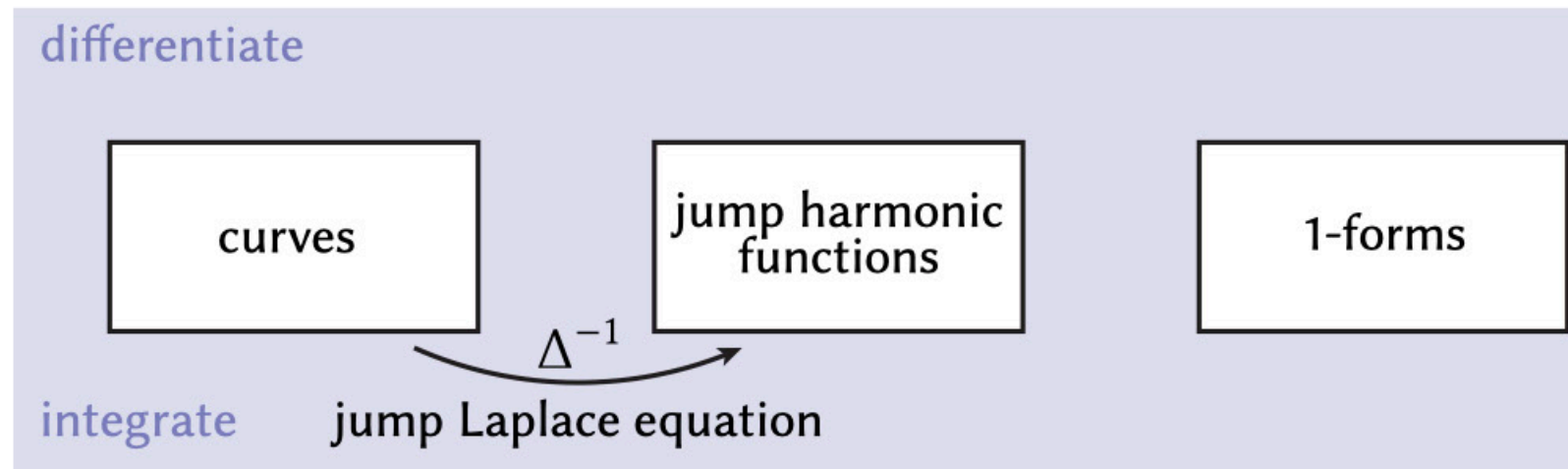
integrate

Input:

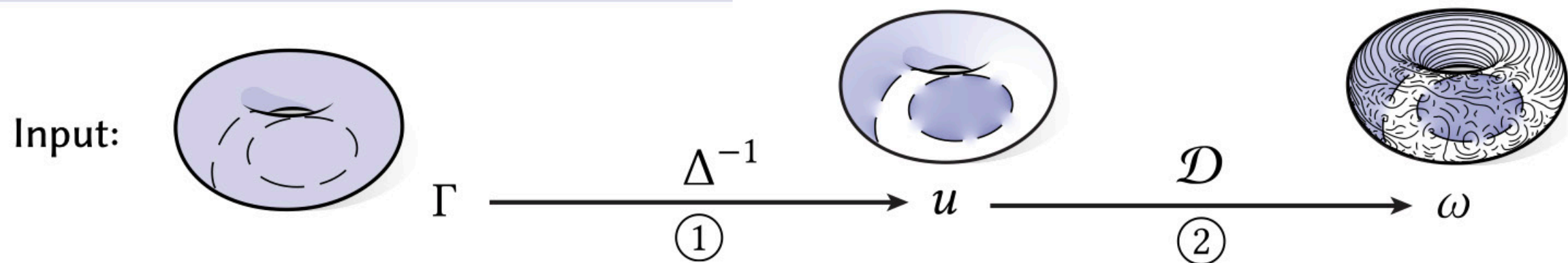
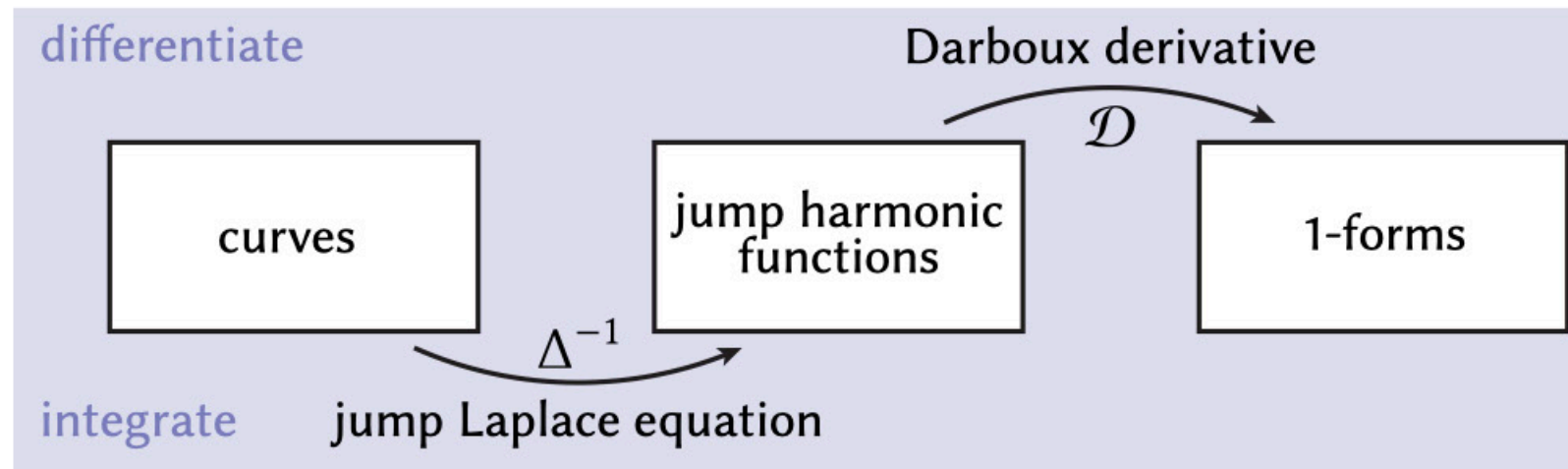


Γ

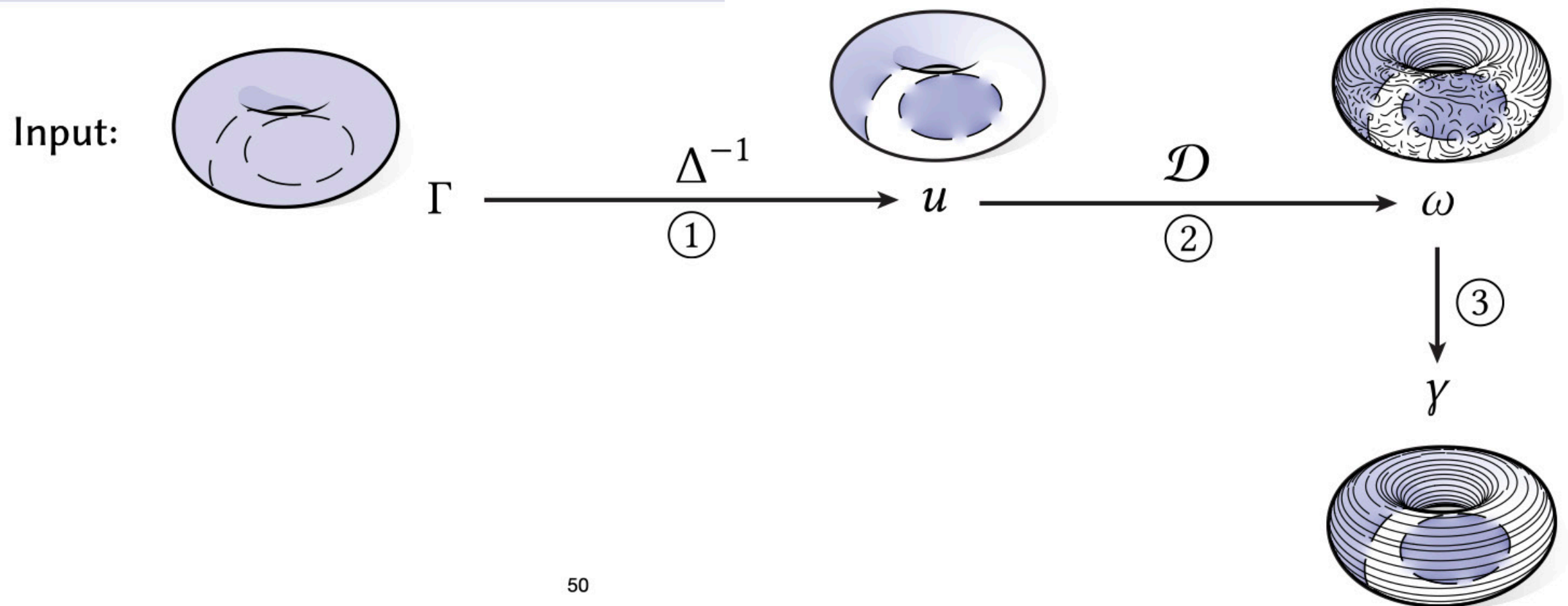
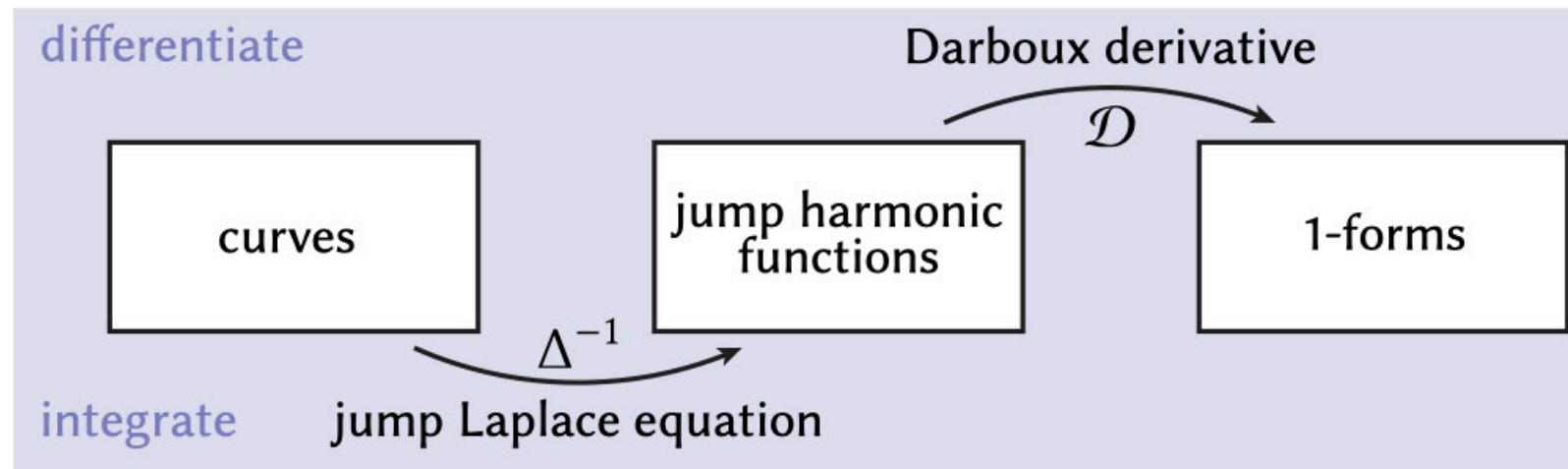
Completing the round-trip



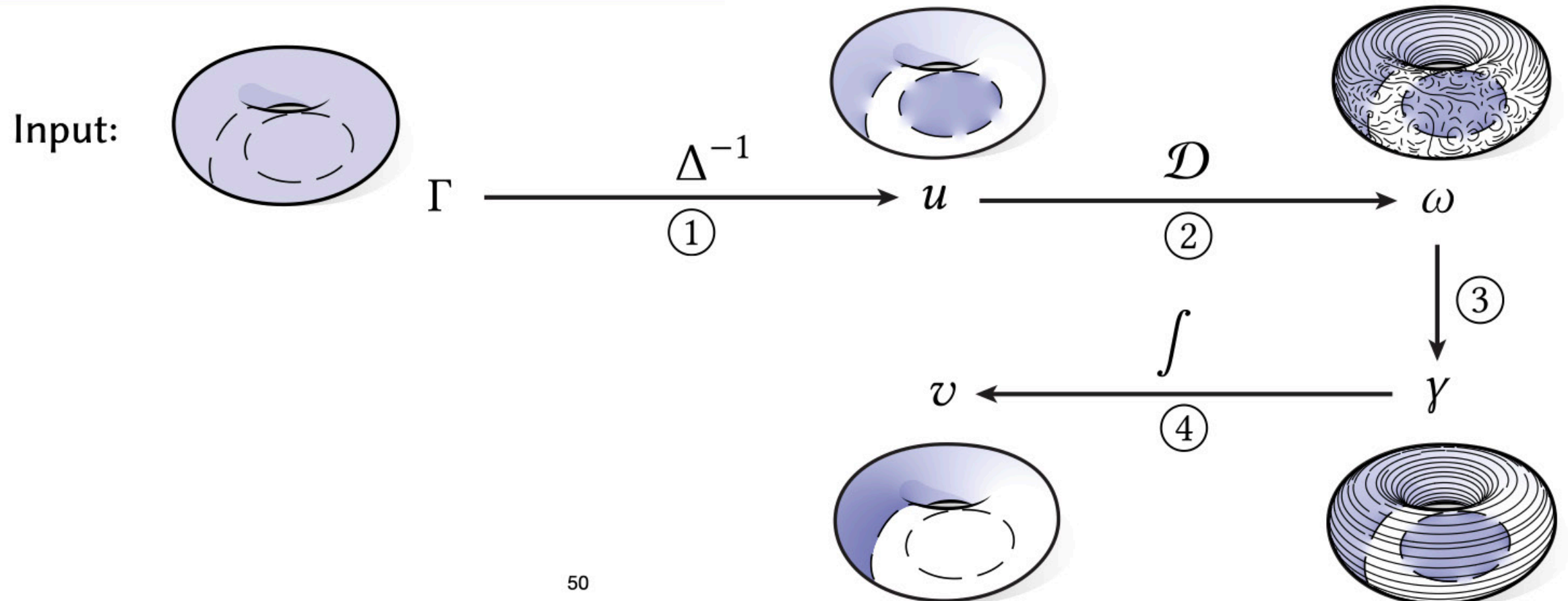
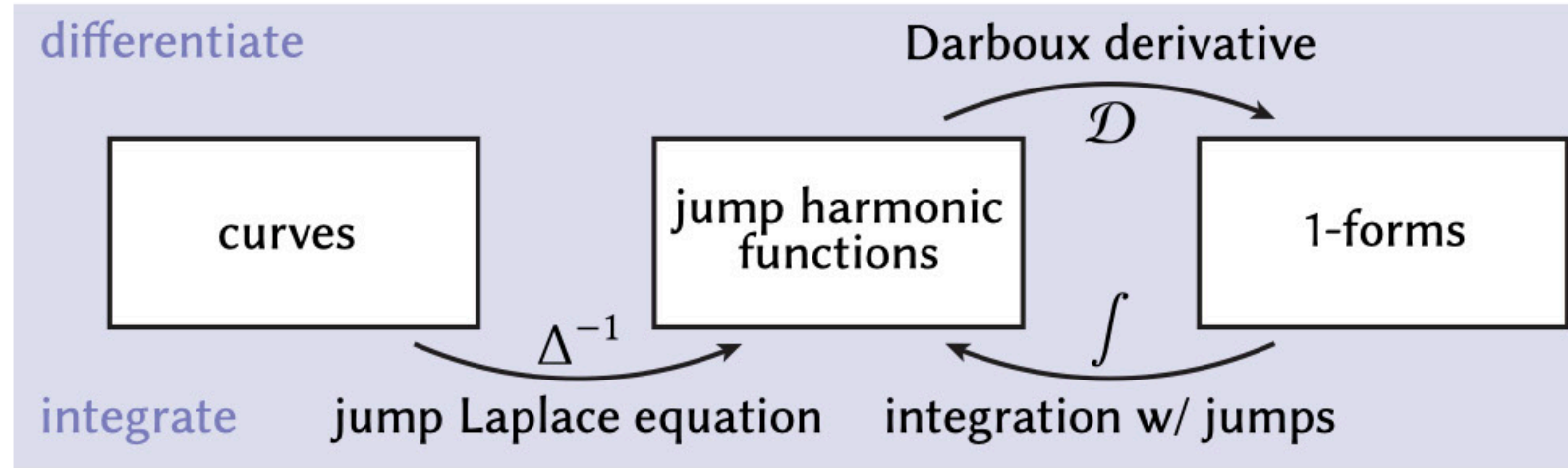
Completing the round-trip



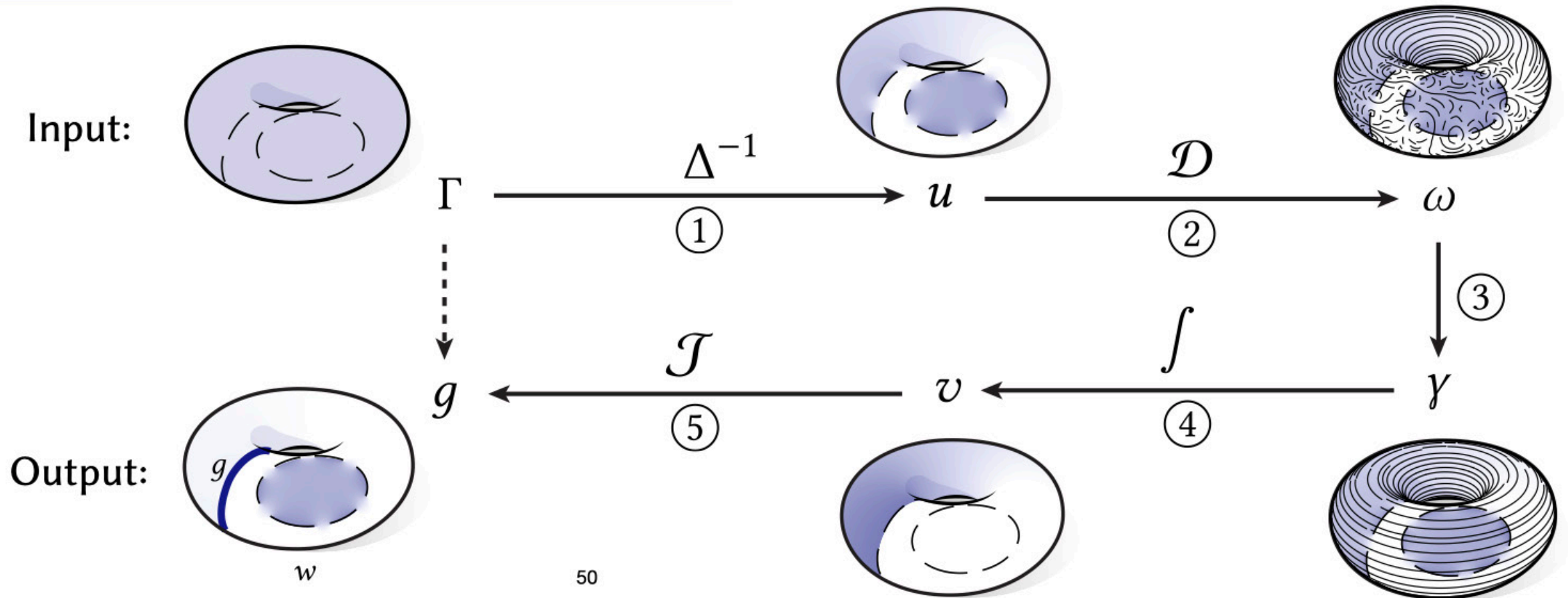
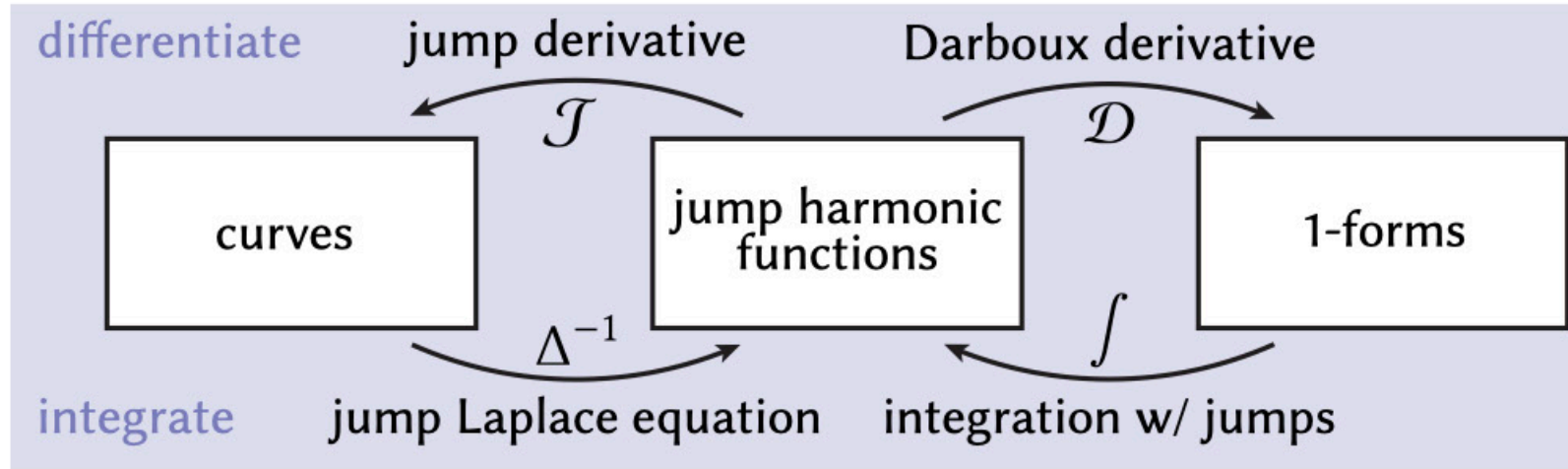
Completing the round-trip



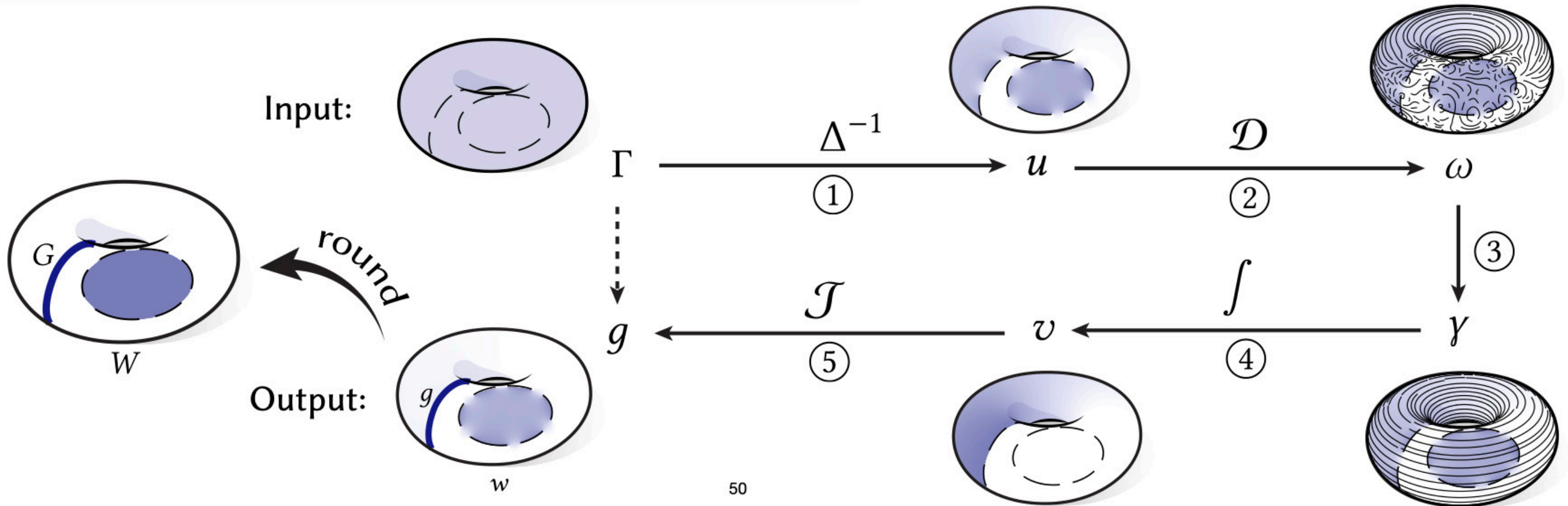
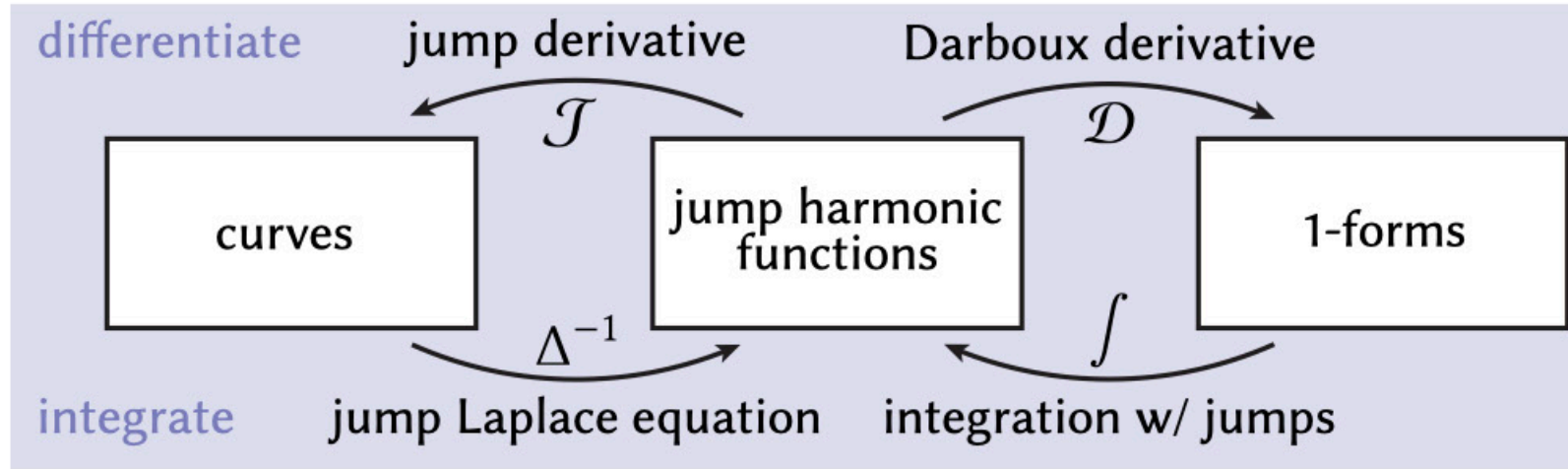
Completing the round-trip



Completing the round-trip



Completing the round-trip



Algorithm summary

Algorithm summary

- Solve for a harmonic function u with jumps Γ .

Algorithm summary

- Solve for a harmonic function u with jumps Γ .
- Compute the harmonic part γ of Du .

Algorithm summary

- Solve for a harmonic function u with jumps Γ .
- Compute the harmonic part γ of $\mathcal{D}u$.
- Solve a (linear) optimization to obtain the residual function v .

Algorithm summary

- Solve for a harmonic function u with jumps Γ .
- Compute the harmonic part γ of $\mathcal{D}u$.
- Solve a (linear) optimization to obtain the residual function v .
- Solve for the winding number function w with jumps $\Gamma - \mathcal{J}v$

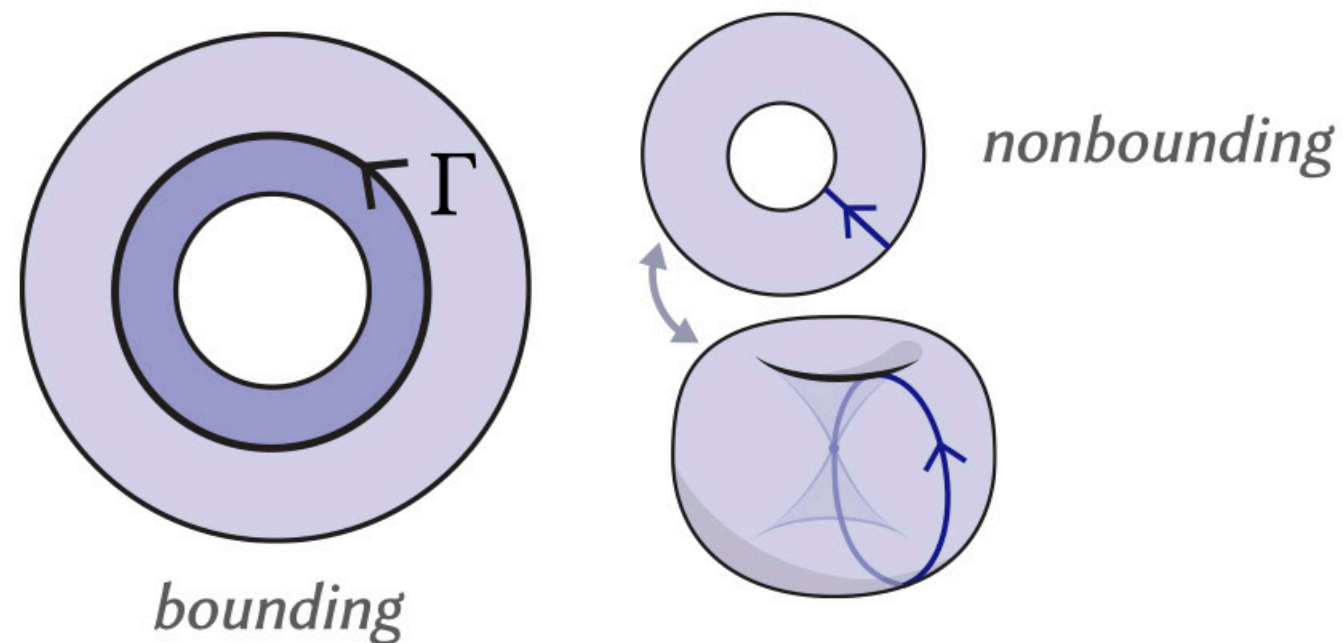
Algorithm summary

- Solve for a harmonic function u with jumps Γ .
- Compute the harmonic part γ of $\mathcal{D}u$.
- Solve a (linear) optimization to obtain the residual function v .
- Solve for the winding number function w with jumps $\Gamma - \mathcal{J}v$

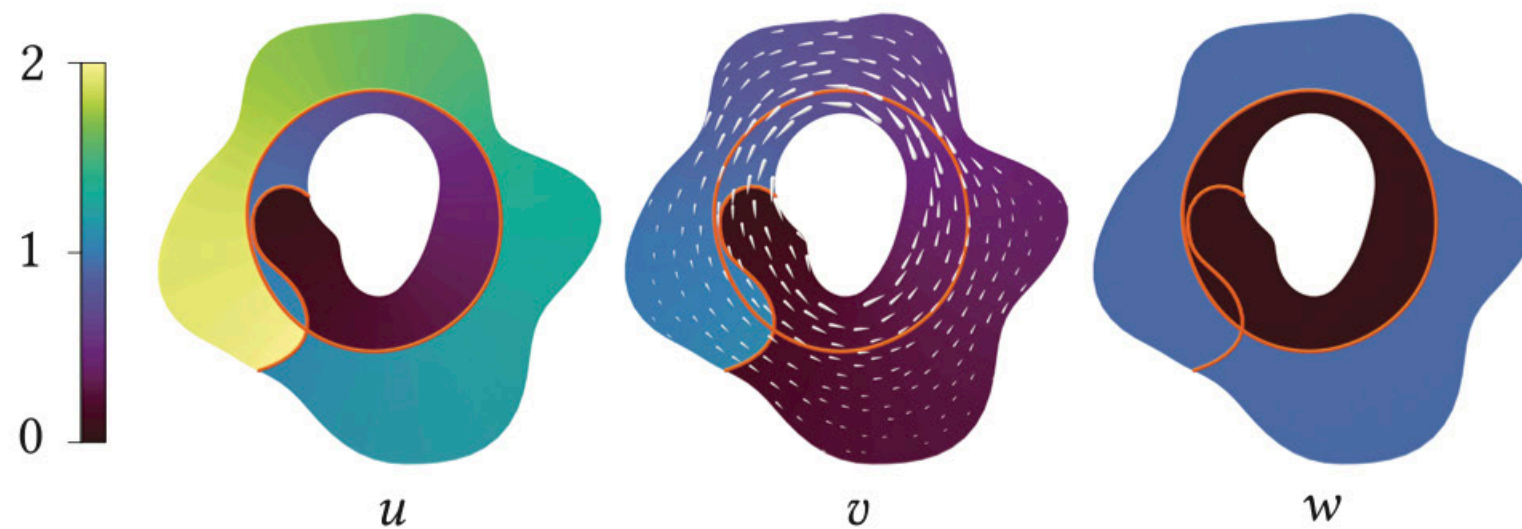
if surface M is multiply-connected

Surfaces with boundary

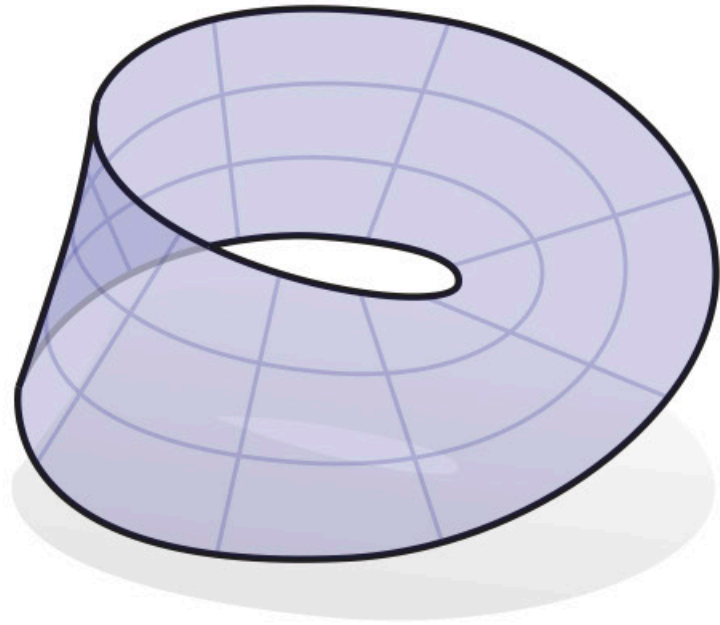
Bounding curves are those congruent to zero in the *relative homology* group $H_1(M, \partial M)$.



The rest of the theory follows.

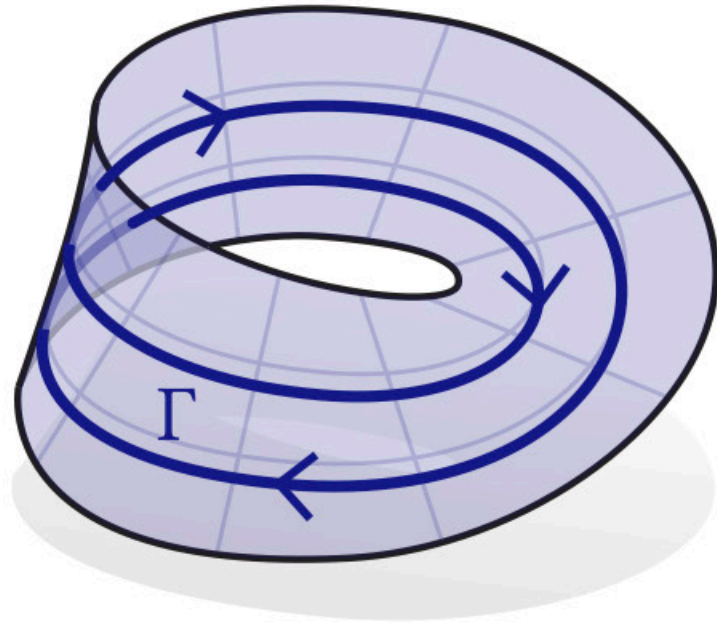


Non-orientable surfaces



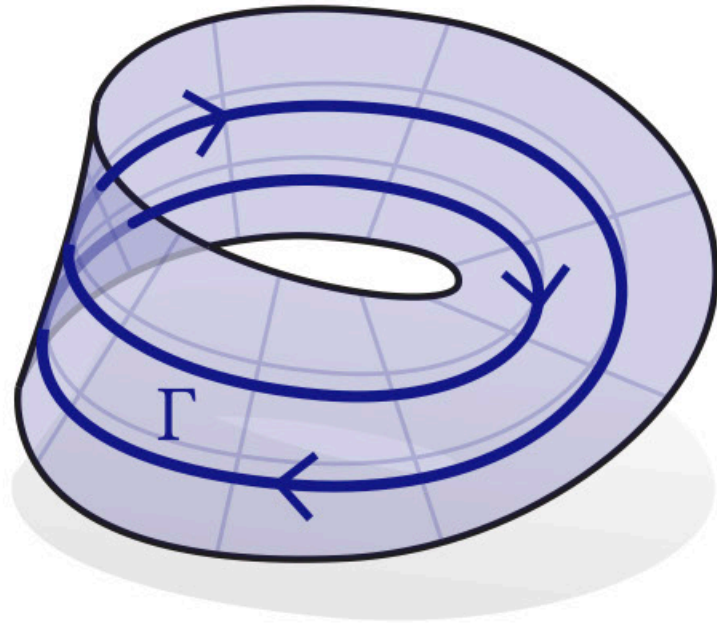
Non-orientable surfaces

orientable curve

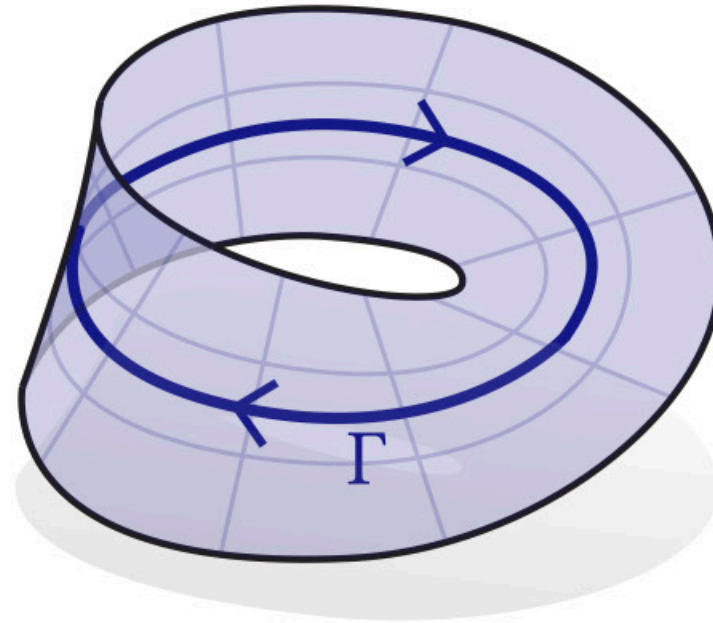


Non-orientable surfaces

orientable curve

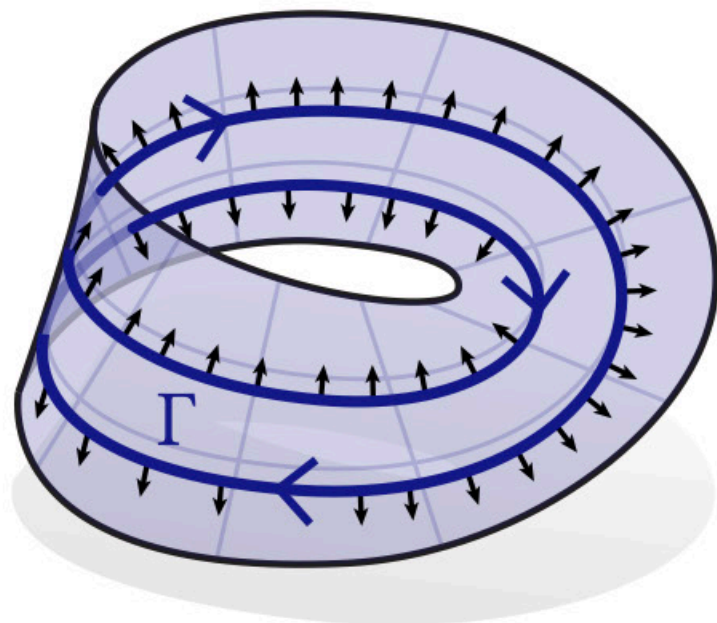


non-orientable curve

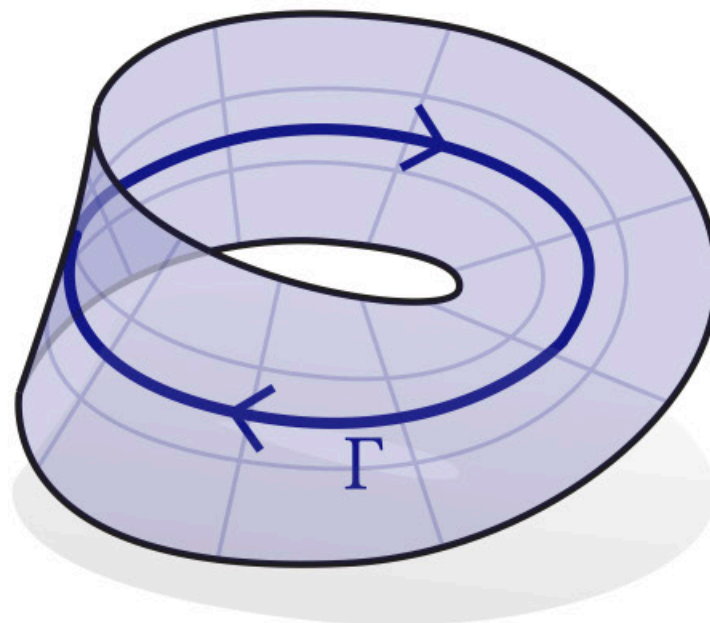


Non-orientable surfaces

orientable curve

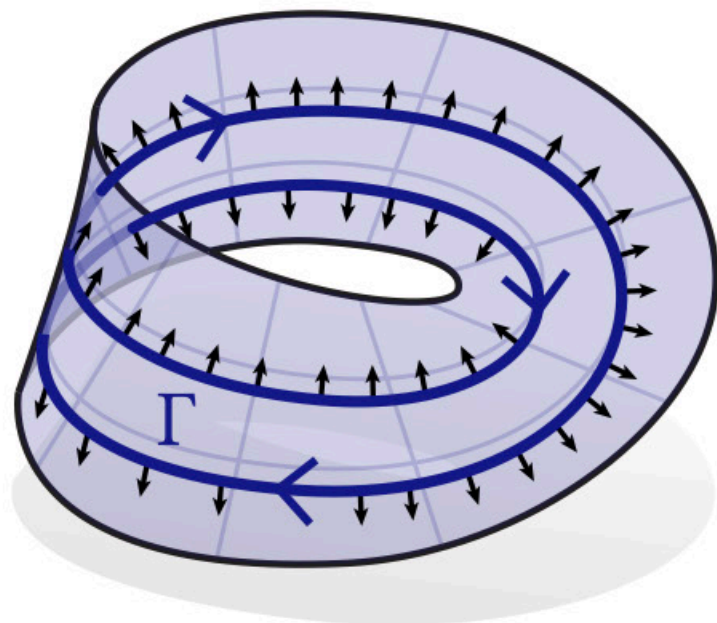


non-orientable curve

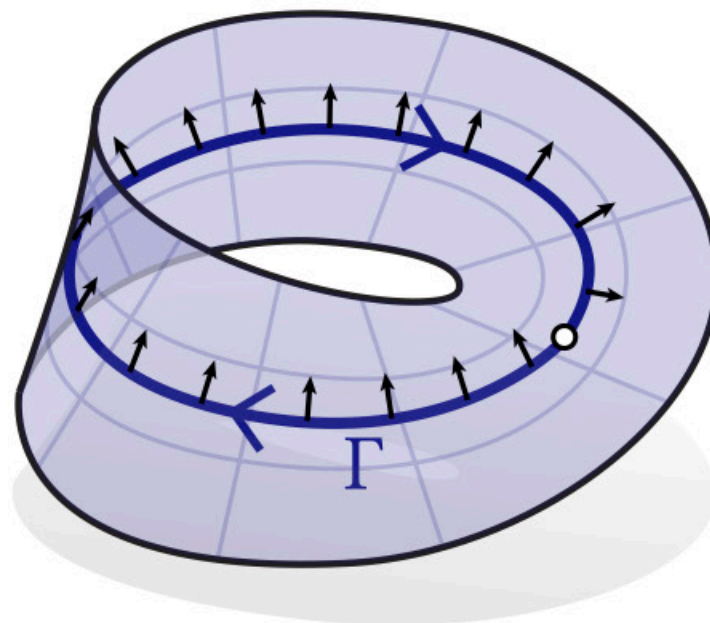


Non-orientable surfaces

orientable curve

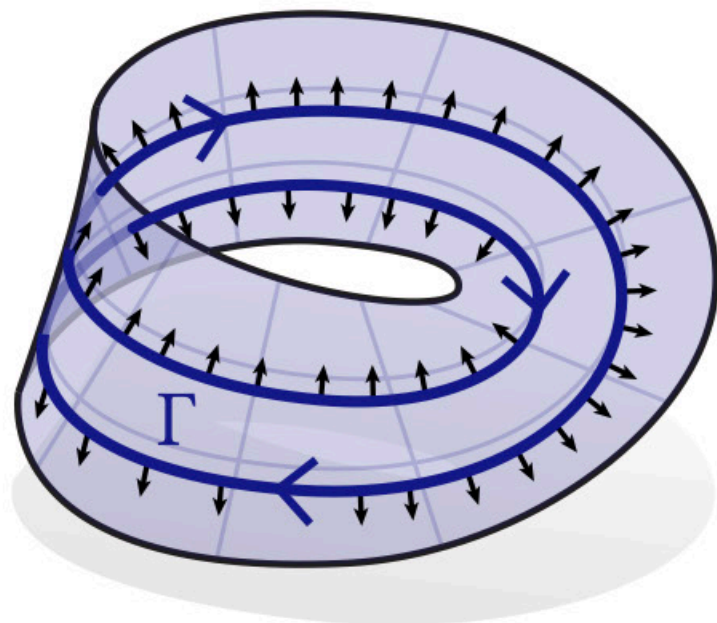


non-orientable curve

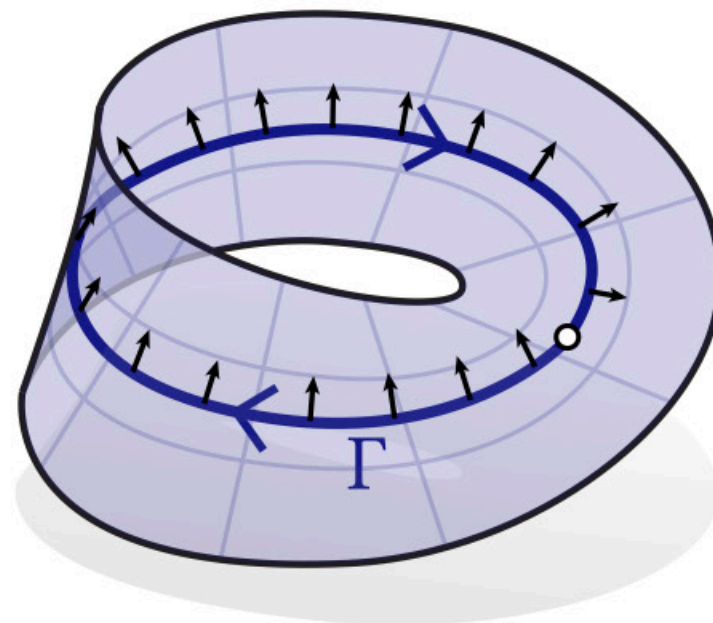


Non-orientable surfaces

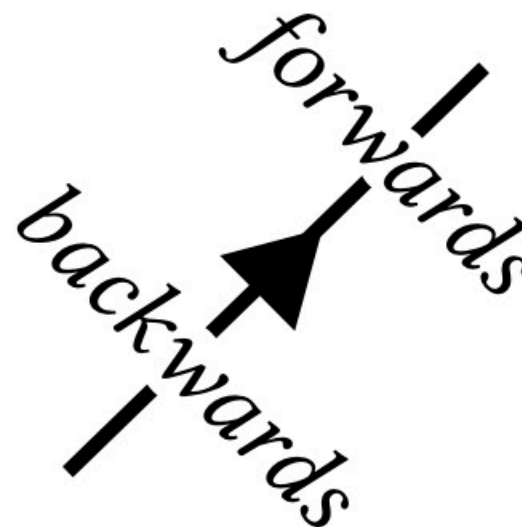
orientable curve



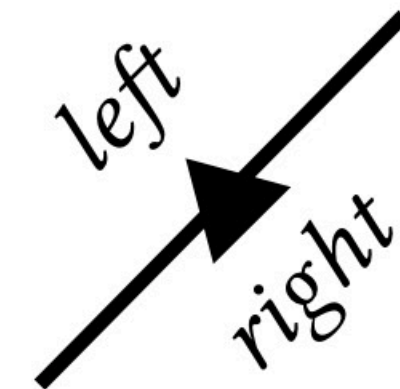
non-orientable curve



tangential

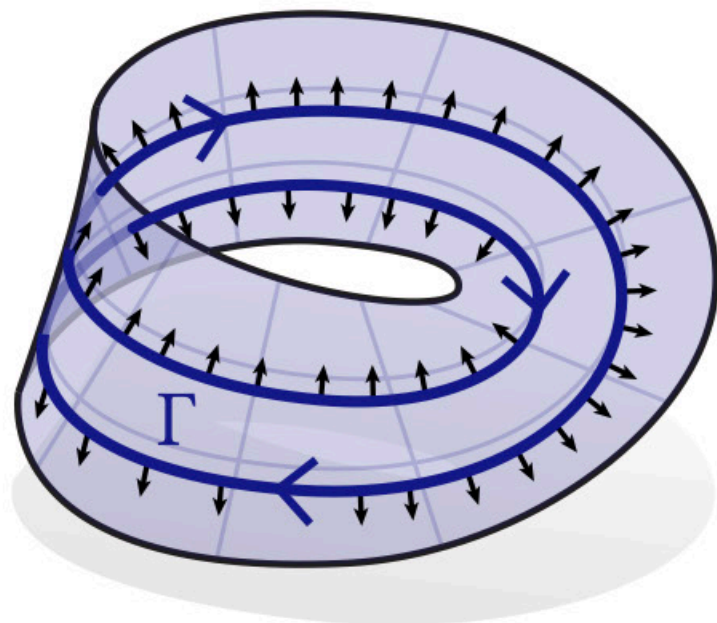


normal

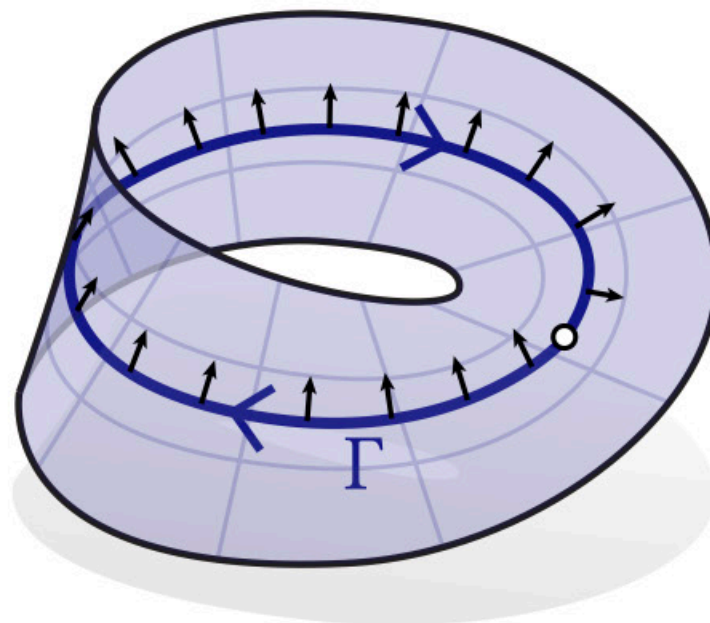


Non-orientable surfaces

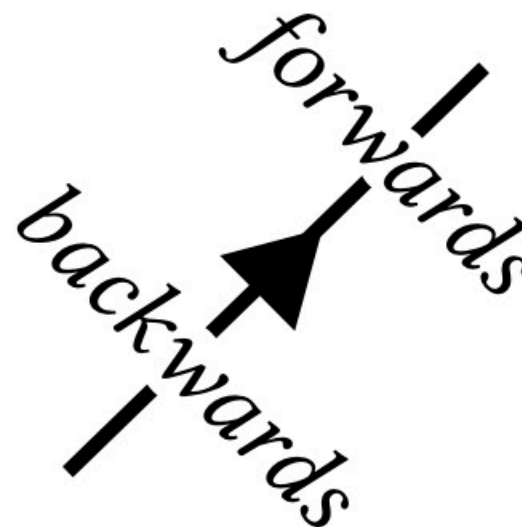
orientable curve



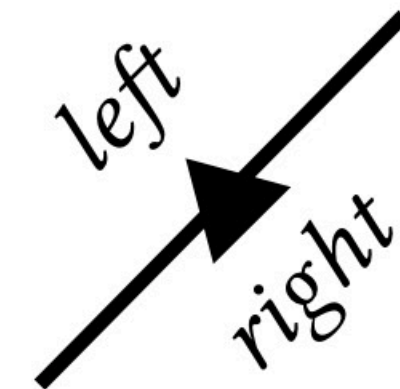
non-orientable curve



tangential



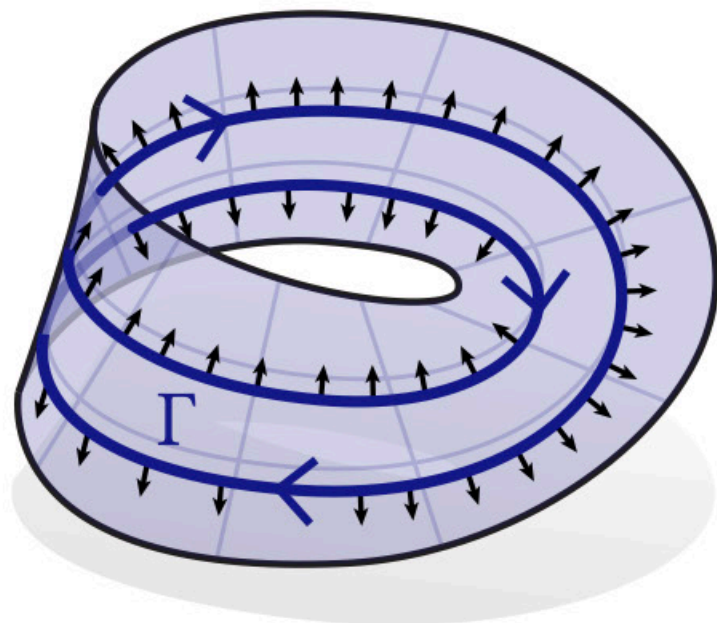
normal



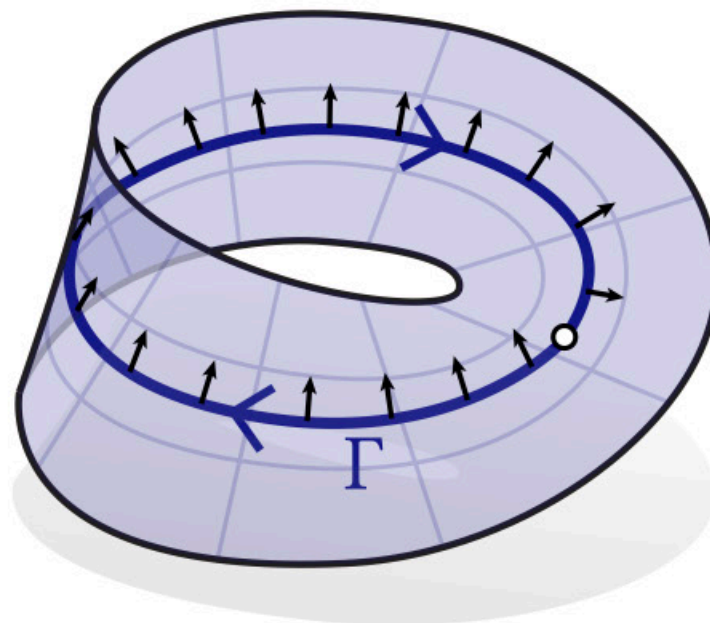
On orientable surfaces,
tangential orientation \equiv normal orientation

Non-orientable surfaces

orientable curve



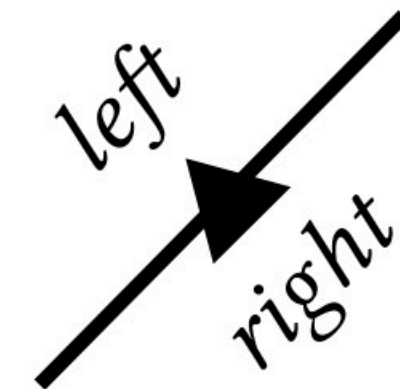
non-orientable curve



tangential



normal

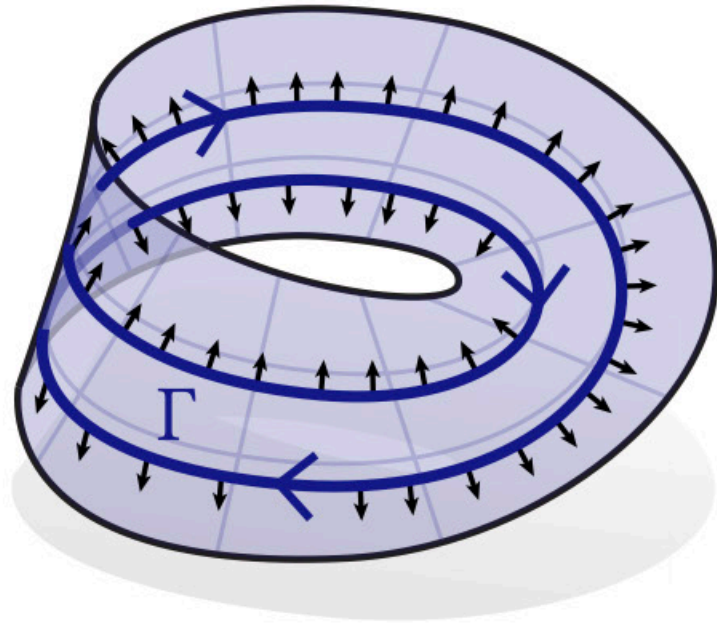


On orientable surfaces,
tangential orientation \equiv normal orientation

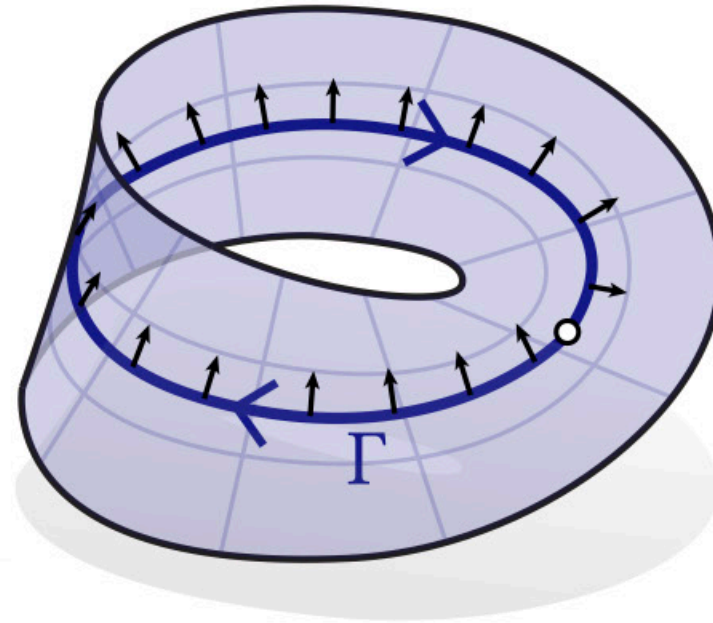
On **non**-orientable surfaces,
must specify normal orientation

Non-orientable surfaces

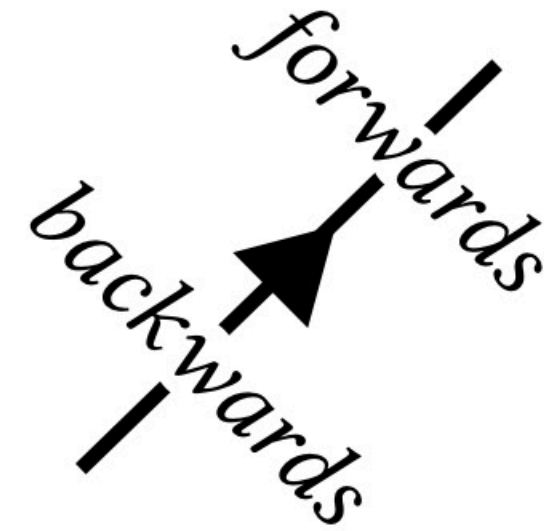
orientable curve



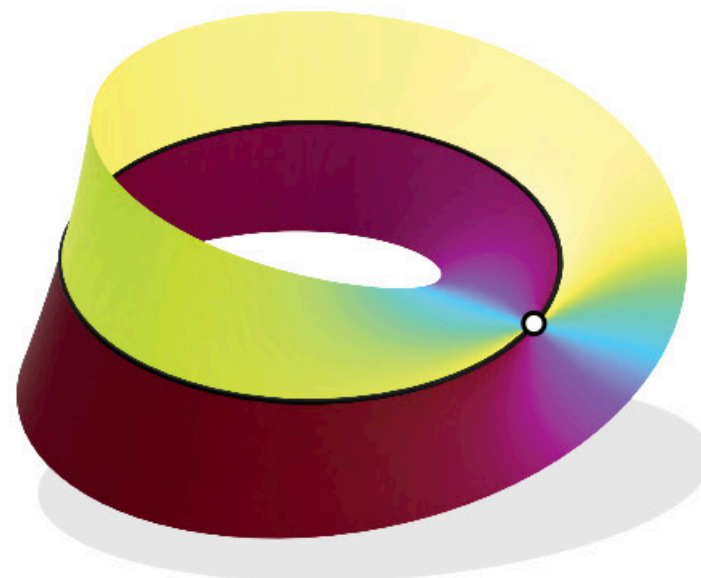
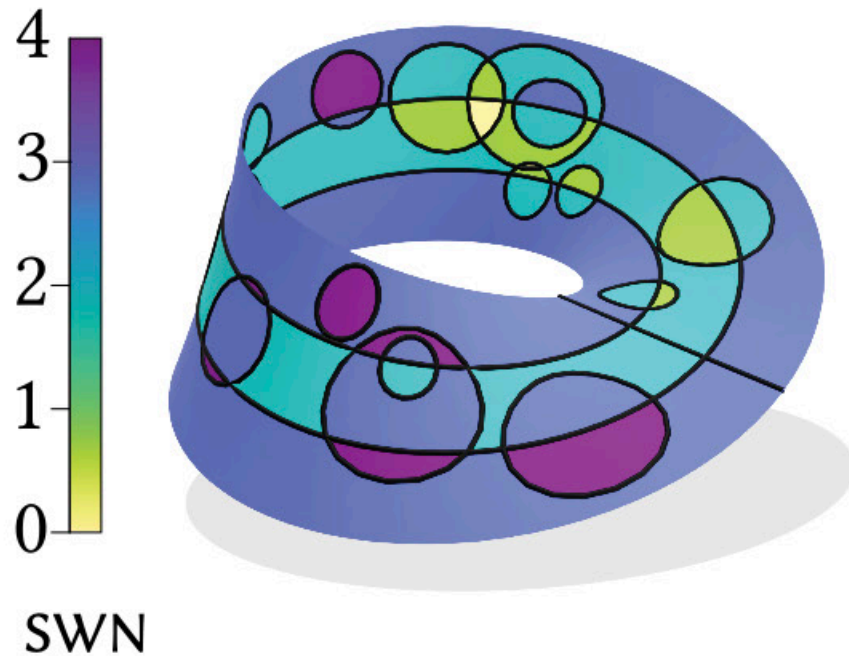
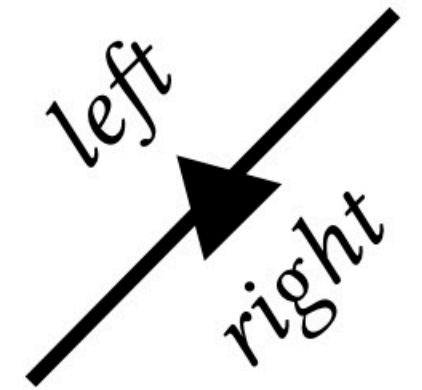
non-orientable curve



tangential



normal

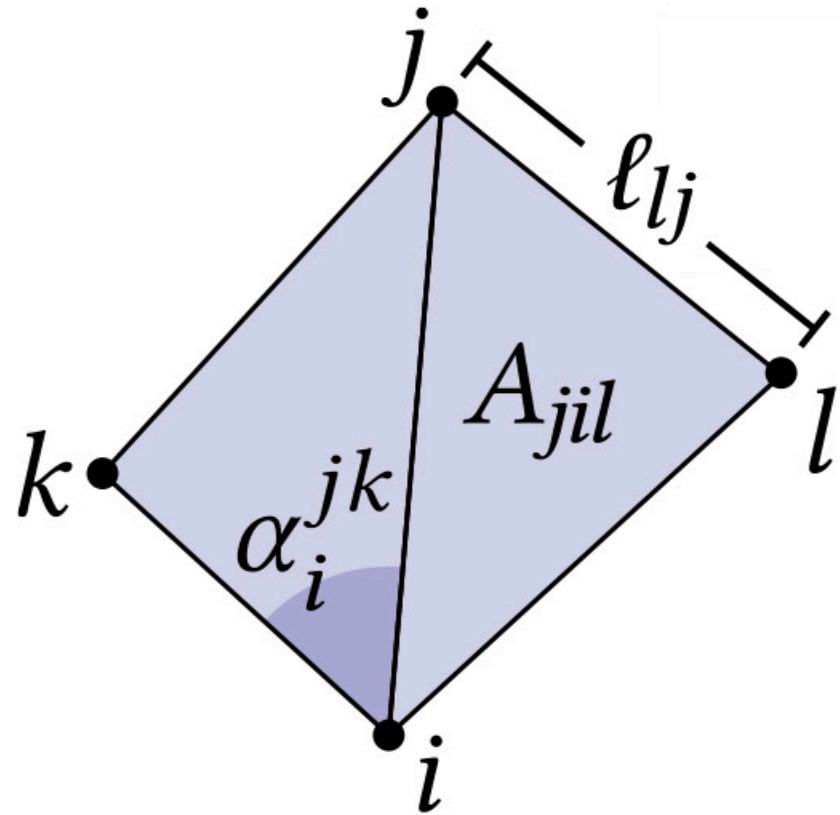


On orientable surfaces,
tangential orientation \equiv normal orientation

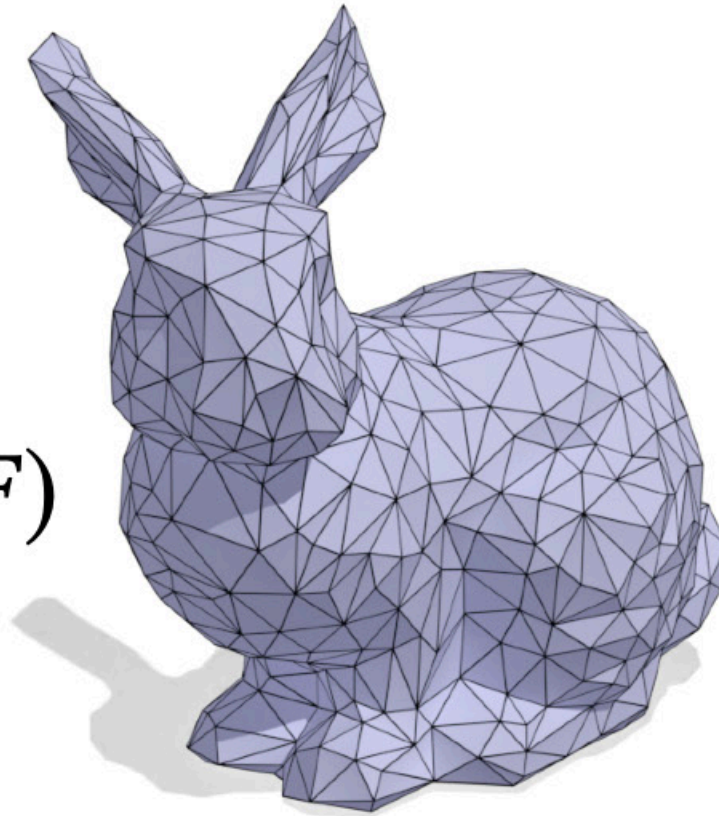
On **non**-orientable surfaces,
must specify normal orientation

DISCRETIZATION

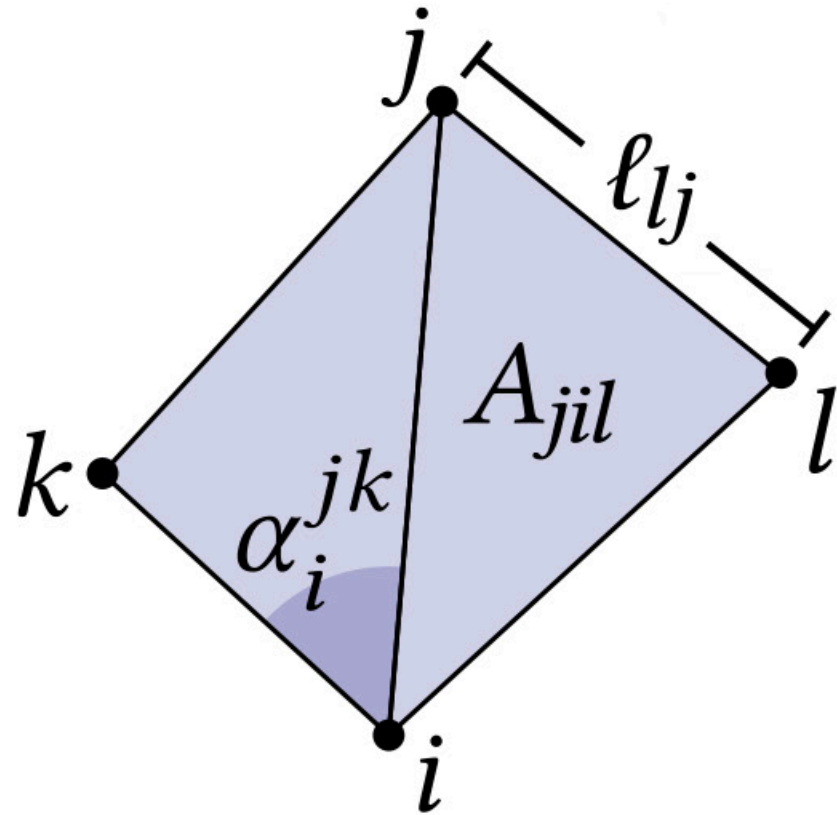
Curves & regions



$$M = (V, E, F)$$

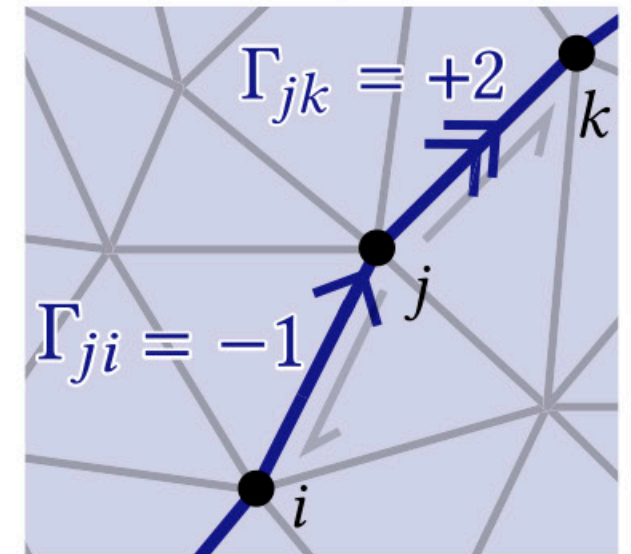
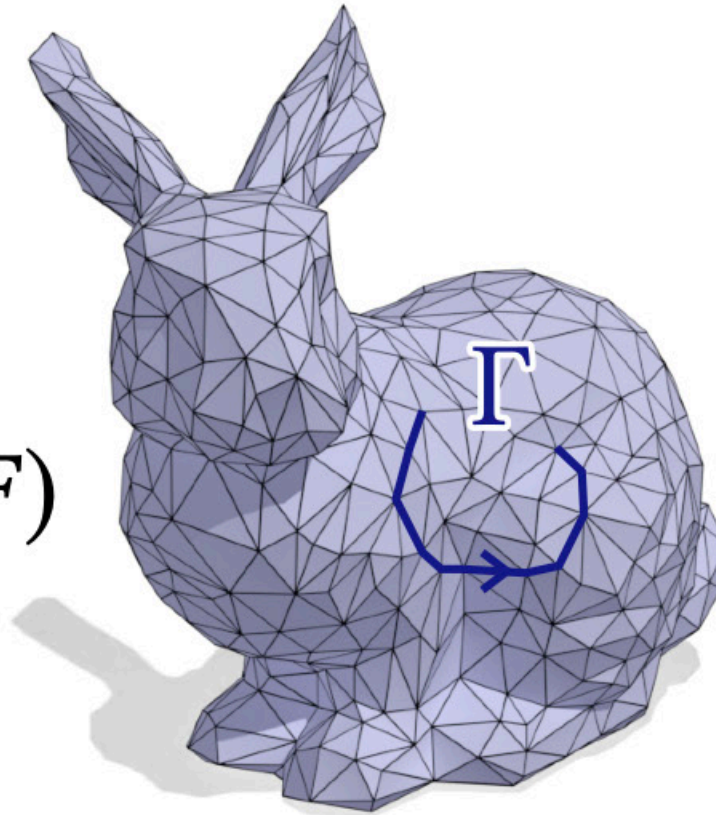


Curves & regions



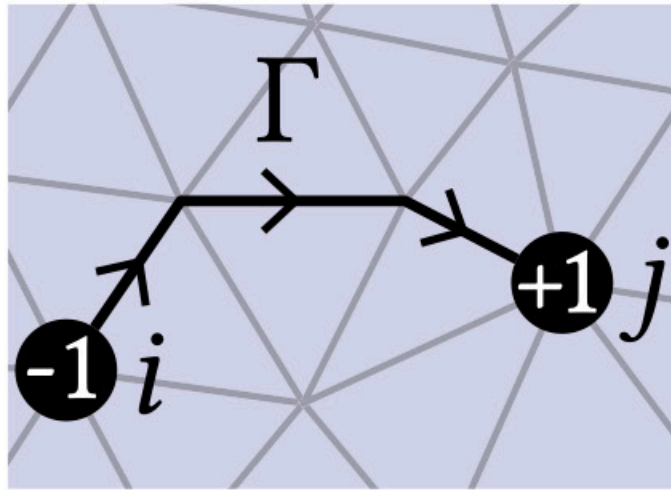
Γ is a **1-chain**, i.e. a signed integer per edge.
Regions are **2-chains**, signed integers per face.

$$M = (V, E, F)$$



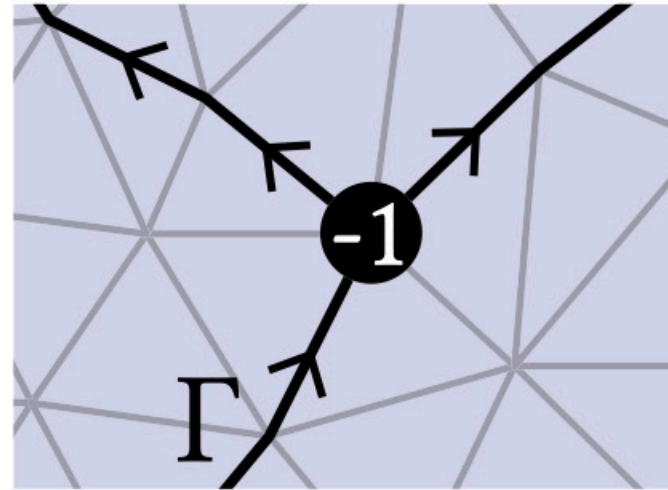
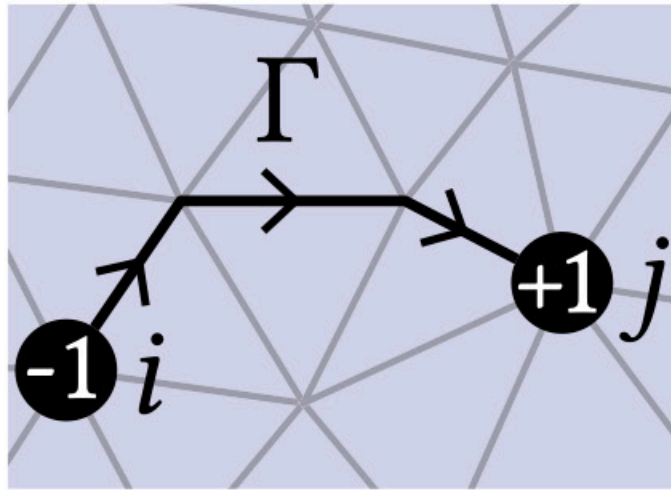
Endpoints

The boundary of Γ is a *0-chain*, $(\partial\Gamma)_i := -\sum_{ij}\Gamma_{ij}$



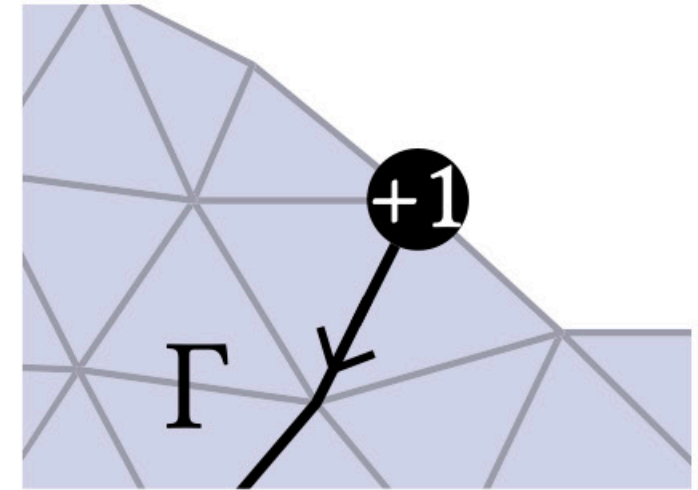
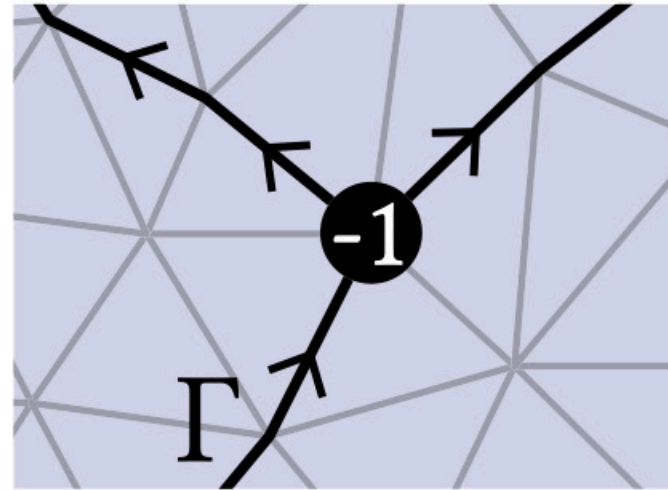
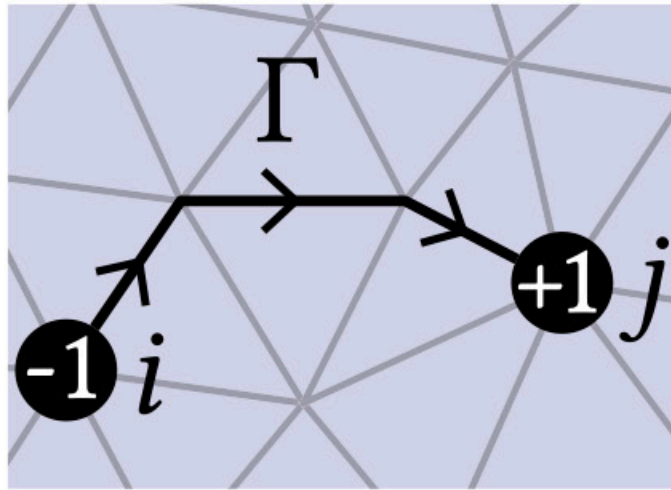
Endpoints

The boundary of Γ is a 0 -chain, $(\partial\Gamma)_i := -\sum_{ij}\Gamma_{ij}$



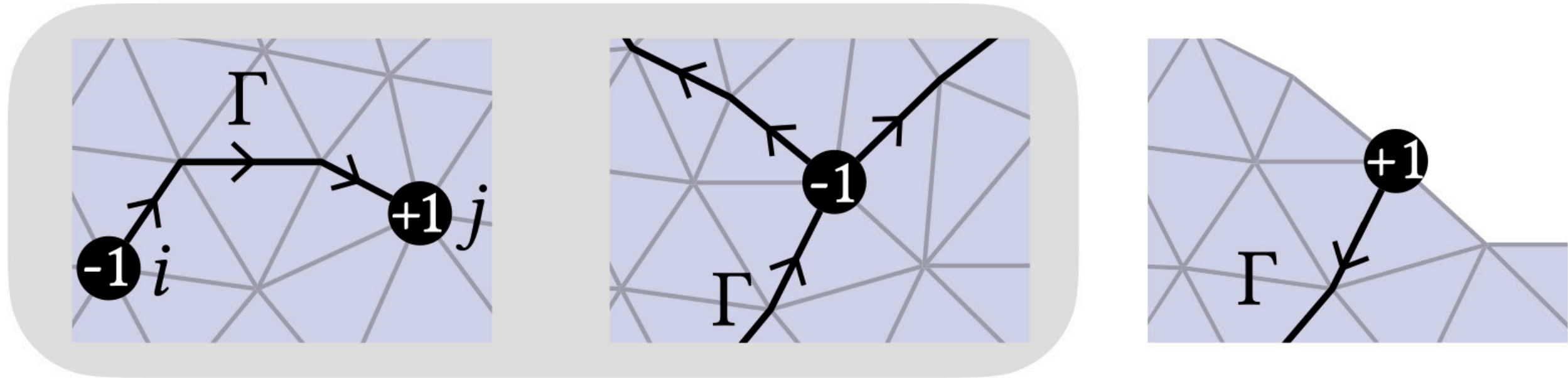
Endpoints

The boundary of Γ is a 0 -chain, $(\partial\Gamma)_i := -\sum_{ij}\Gamma_{ij}$



Endpoints

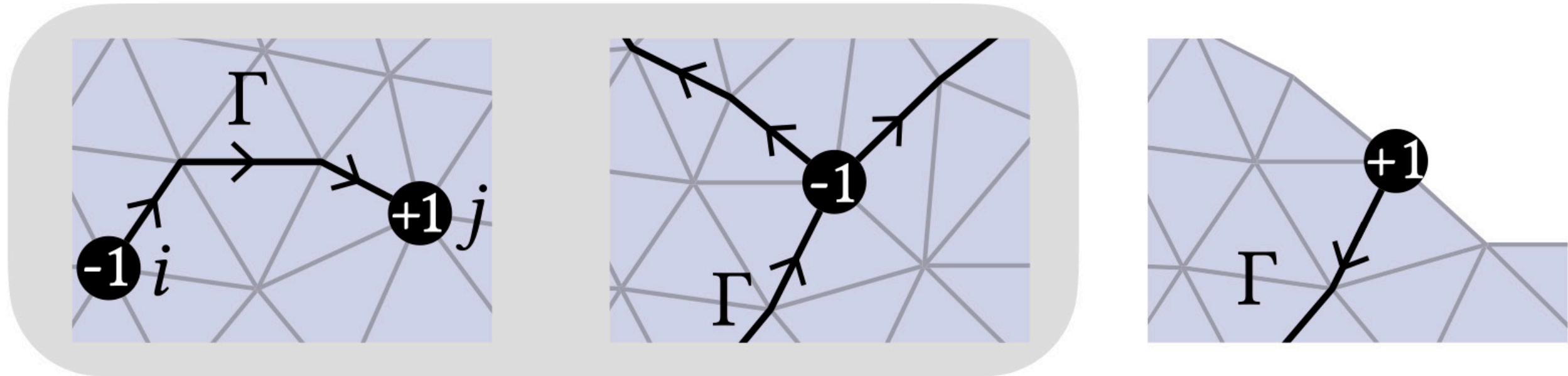
The boundary of Γ is a *0-chain*, $(\partial\Gamma)_i := -\sum_{ij}\Gamma_{ij}$



interior endpoints

Endpoints

The boundary of Γ is a 0 -chain, $(\partial\Gamma)_i := -\sum_{ij}\Gamma_{ij}$

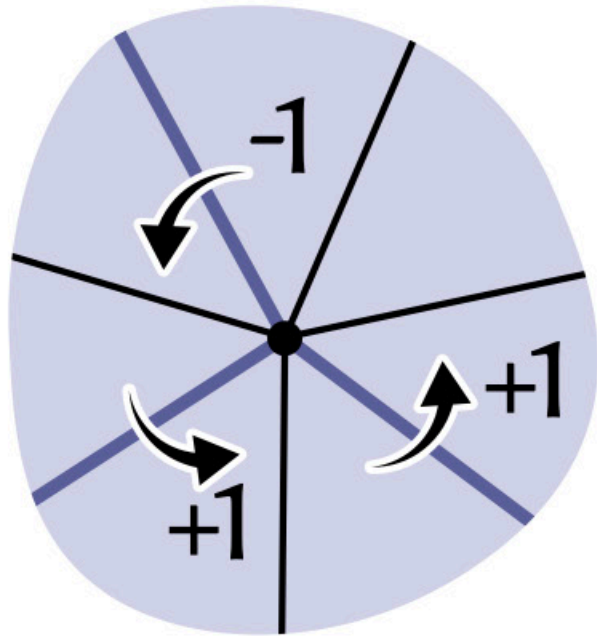


interior endpoints

V^* := set of mesh vertices that are not interior endpoints

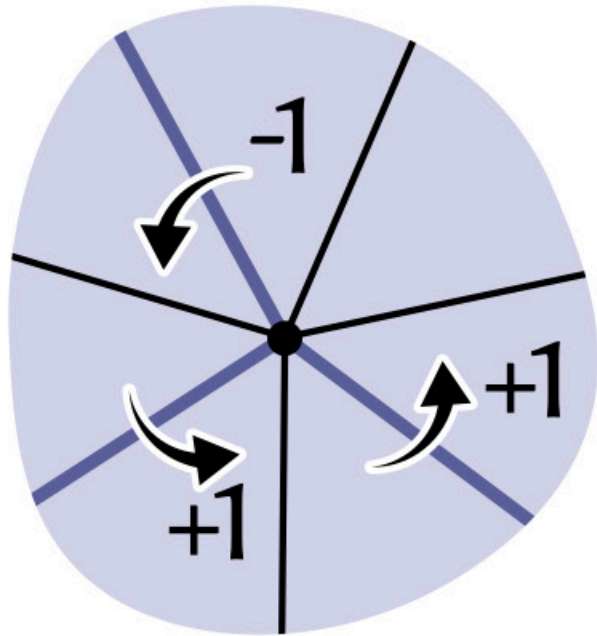
E^* := set of edges with both points in V^*

Endpoints are singular



Endpoints represent *singular points*:
There are no corner values compatible with jumps.

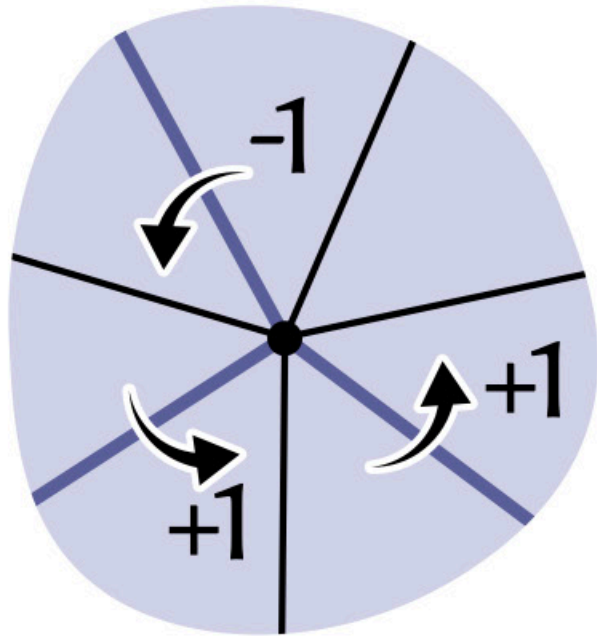
Endpoints are singular



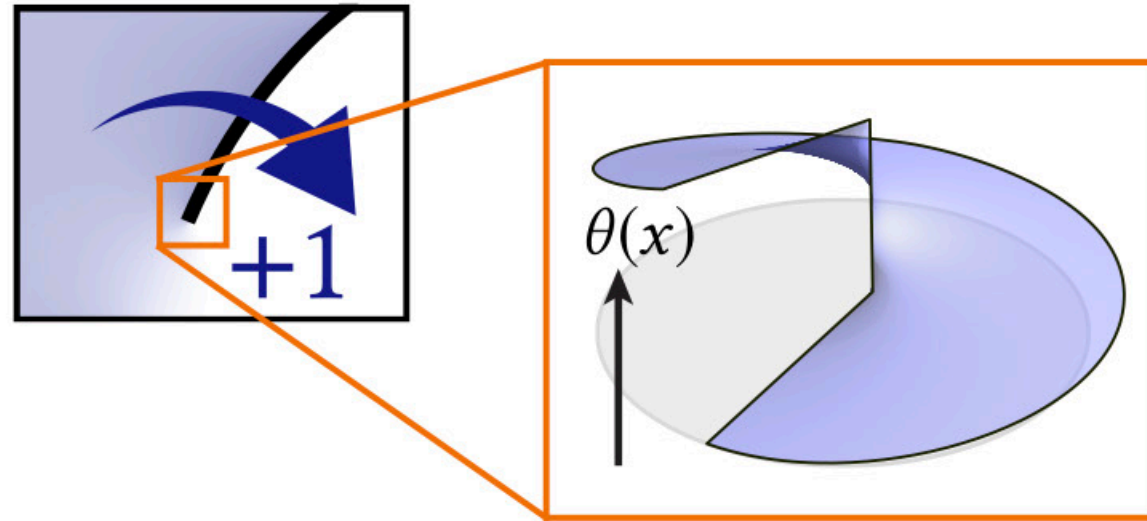
Endpoints represent *singular points*:
There are no corner values compatible with jumps.



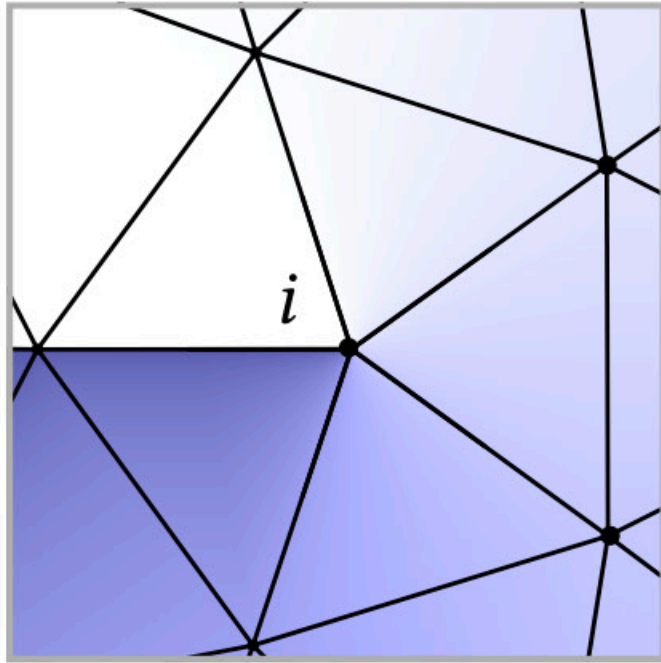
Endpoints are singular



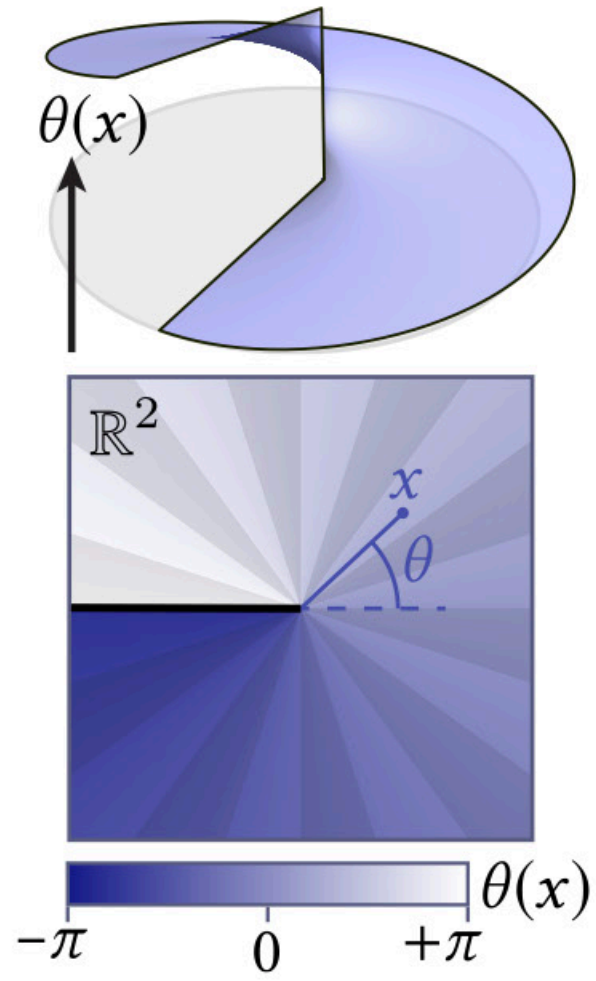
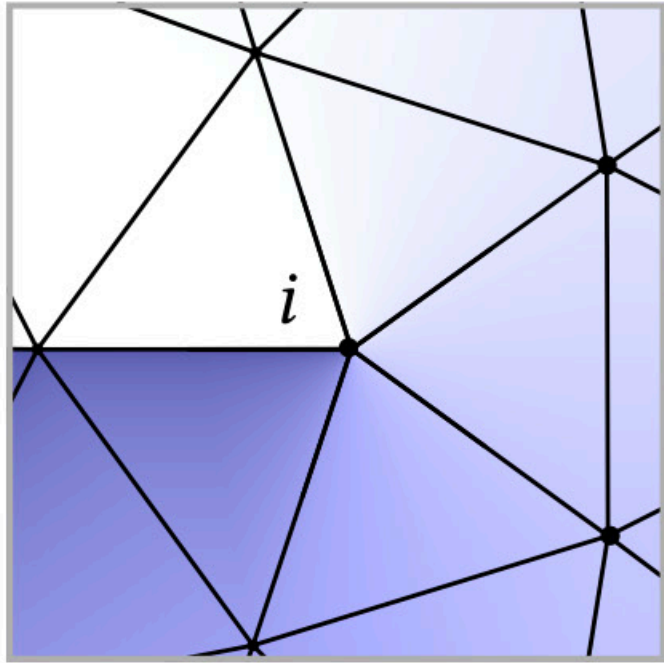
Endpoints represent *singular points*:
There are no corner values compatible with jumps.



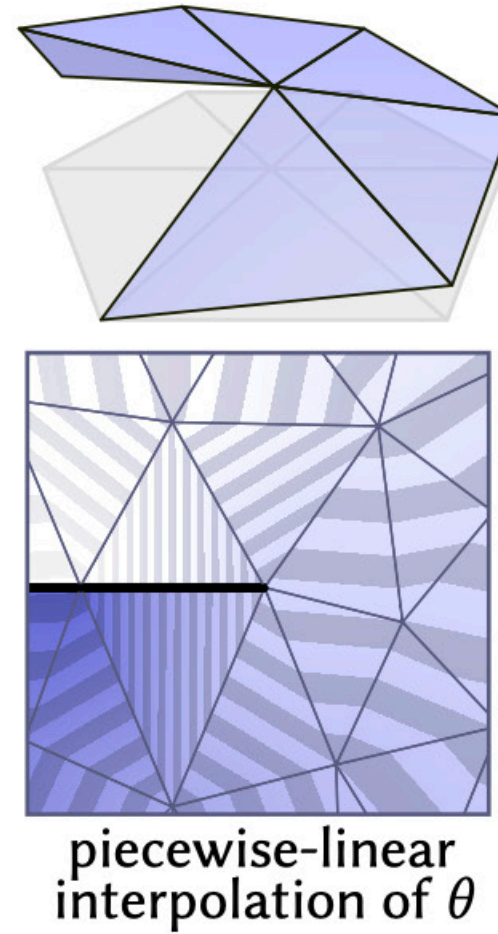
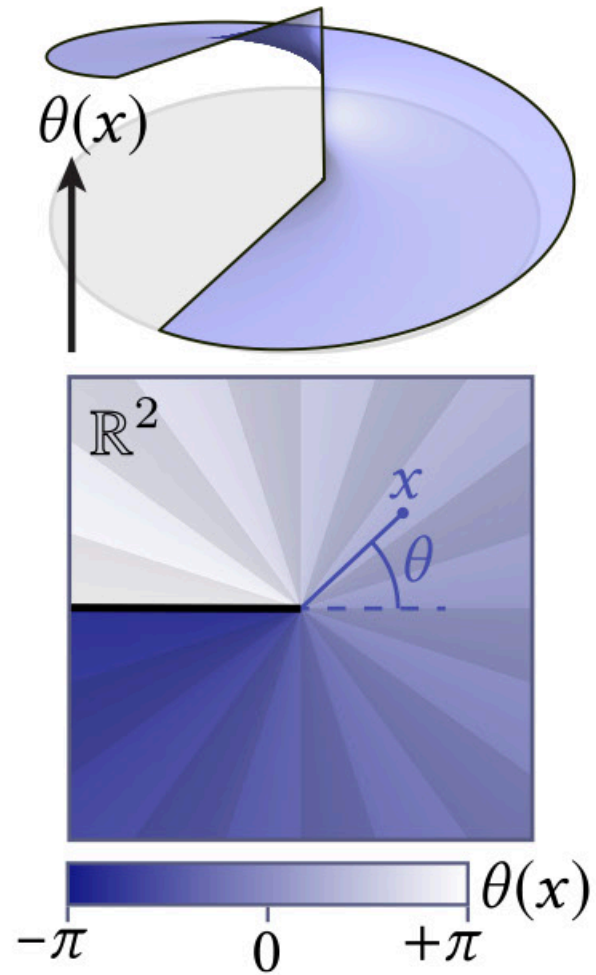
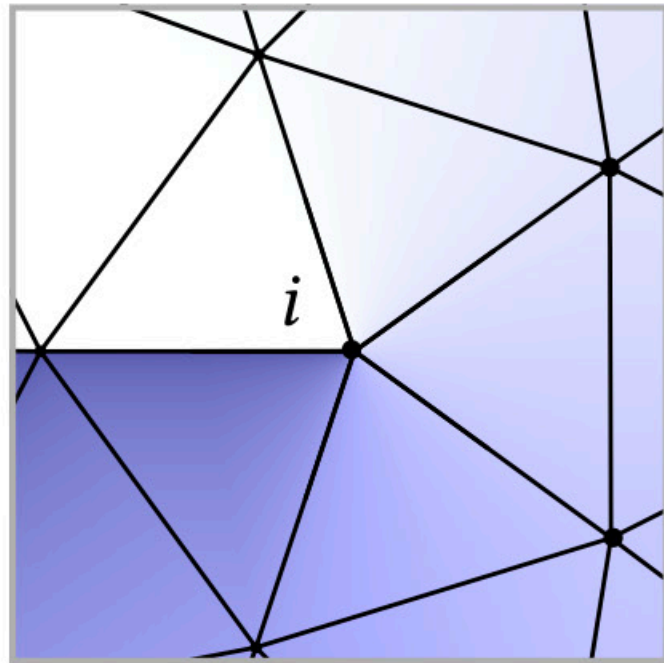
Singular point interpolation



Singular point interpolation

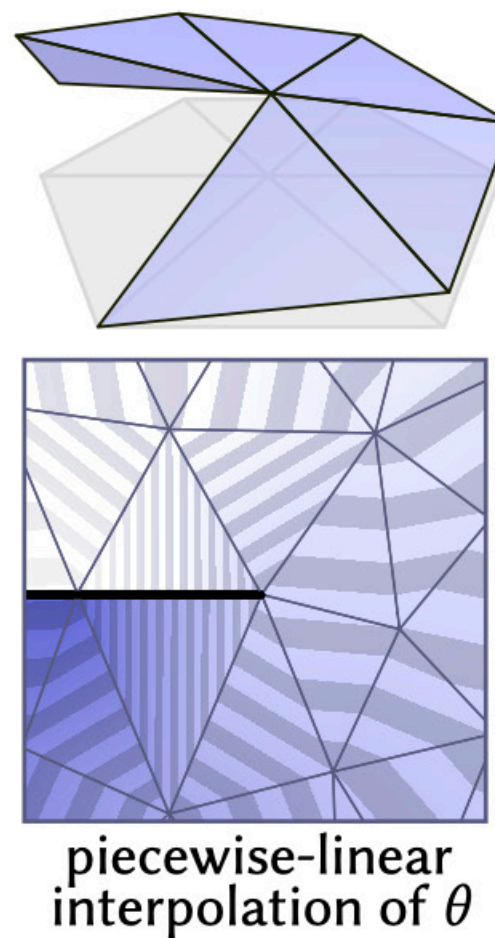
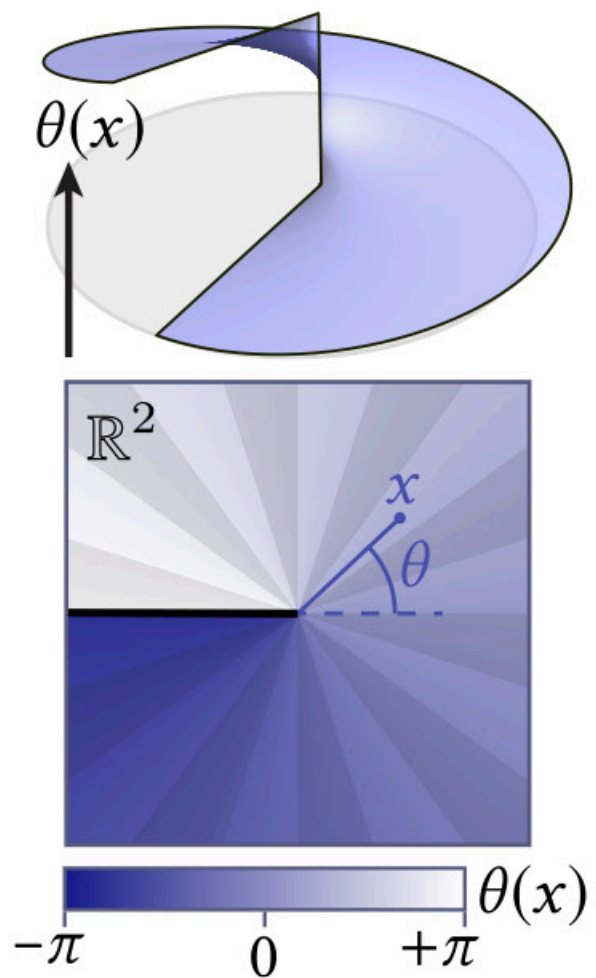
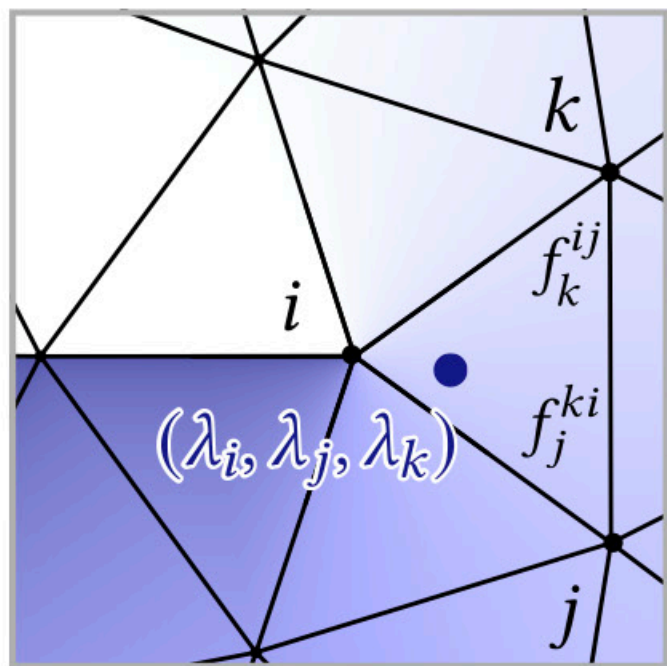


Singular point interpolation



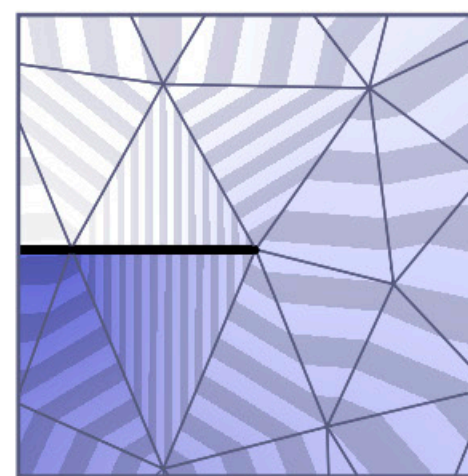
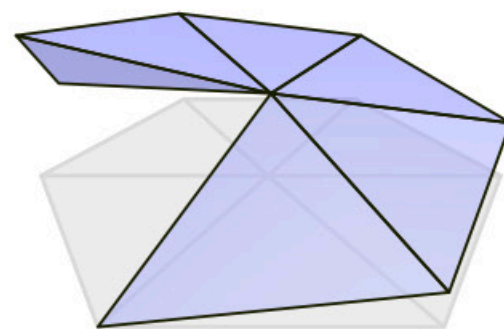
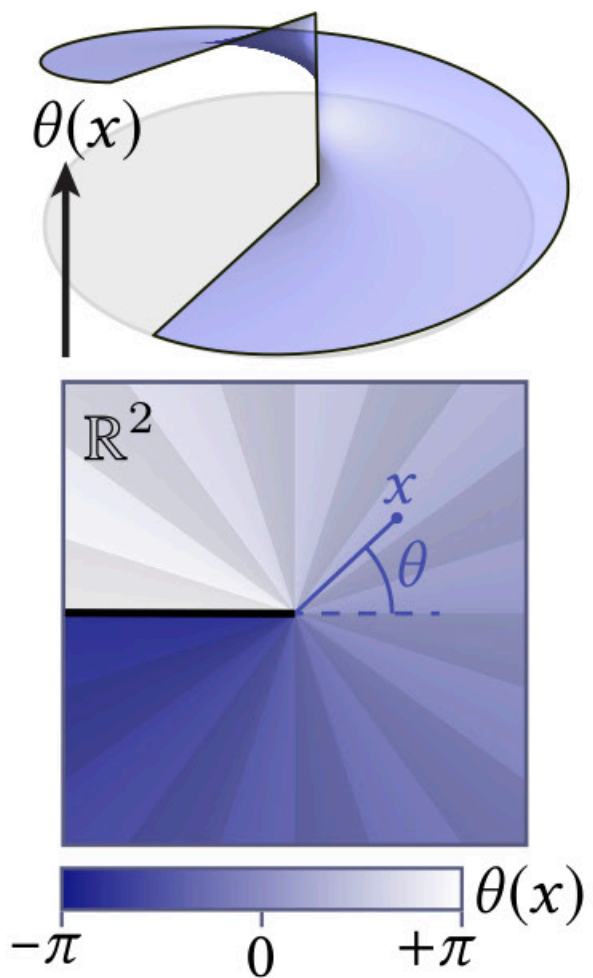
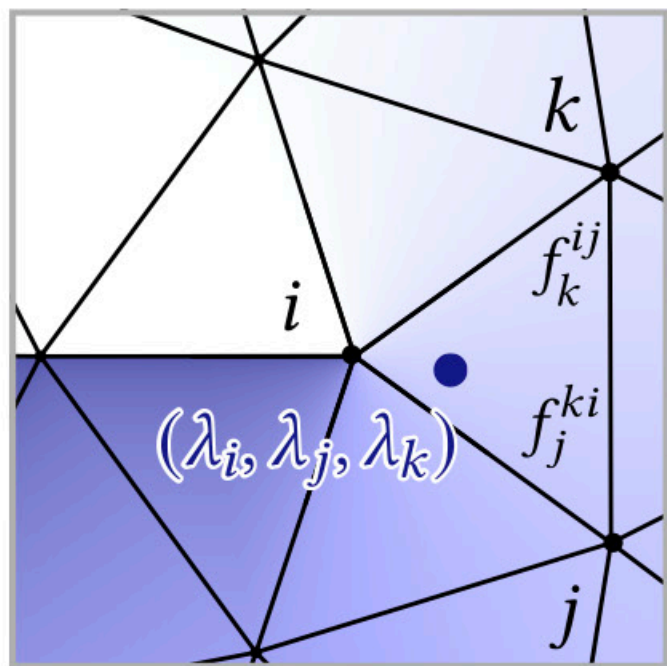
Singular point interpolation

$$f(\lambda_i, \lambda_j, \lambda_k) := \frac{\lambda_j f_j^{ki} + \lambda_k f_k^{ij}}{\lambda_j + \lambda_k}$$

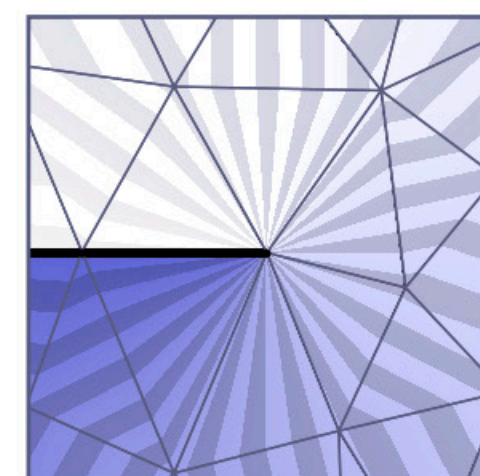
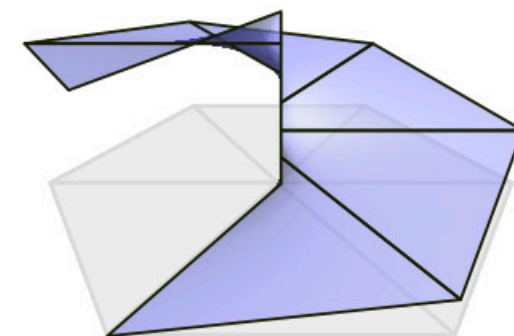


Singular point interpolation

$$f(\lambda_i, \lambda_j, \lambda_k) := \frac{\lambda_j f_j^{ki} + \lambda_k f_k^{ij}}{\lambda_j + \lambda_k}$$

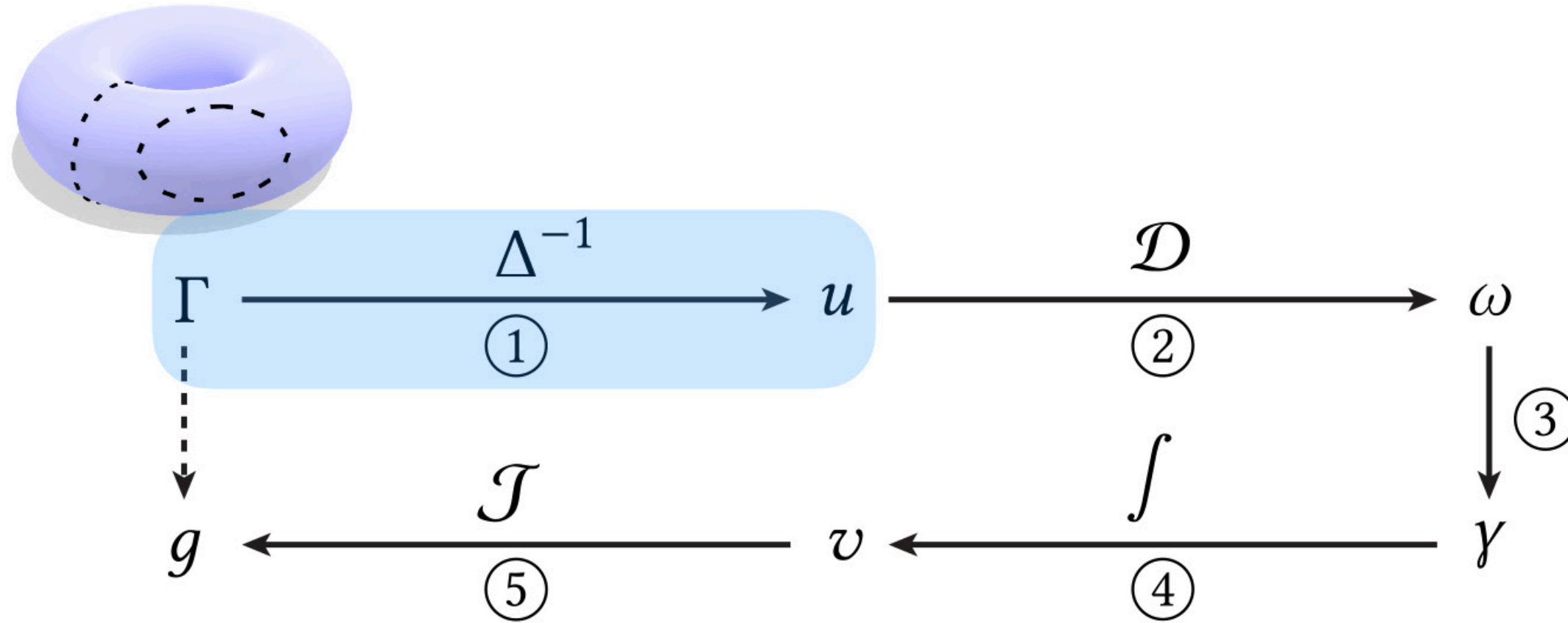


piecewise-linear interpolation of θ

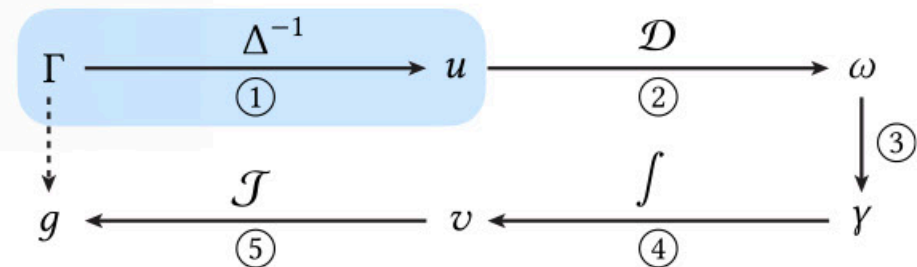


ours

Solving the jump Laplace equation...



The discrete jump Laplacian

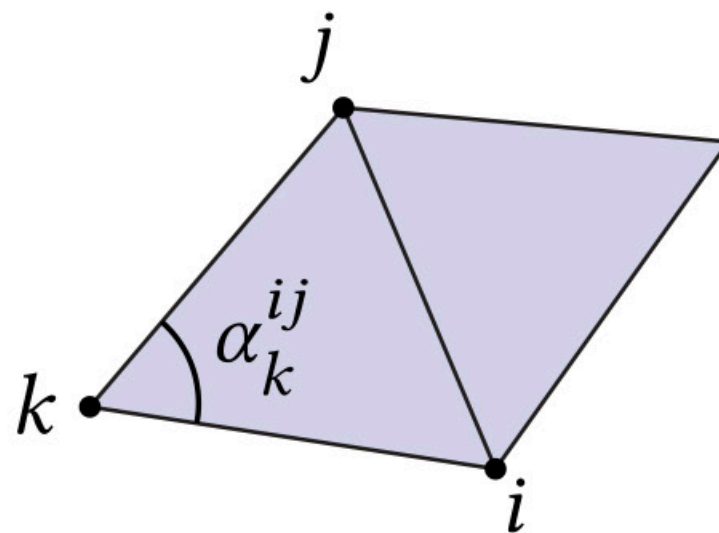


Build the standard cotan Laplacian on V^* :

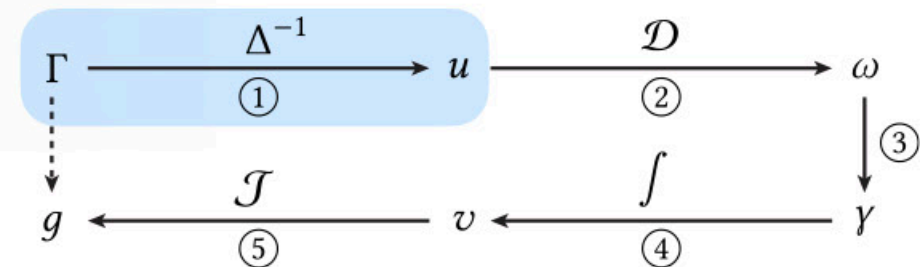
$$L_{ij} = L_{ji} = -w_{ij}, \quad \forall ij \in E^*$$

$$L_{ii} = \sum_{ij \in E^*} w_{ij}, \quad \forall i \in V^*$$

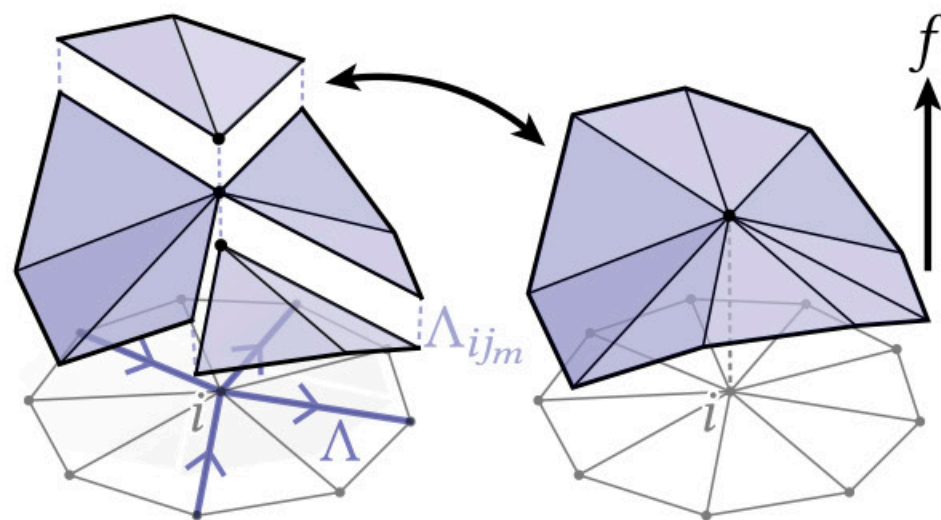
$$w_{ij} := \frac{1}{2} \sum_{ijk \in F} \cot \alpha_k^{ij}$$



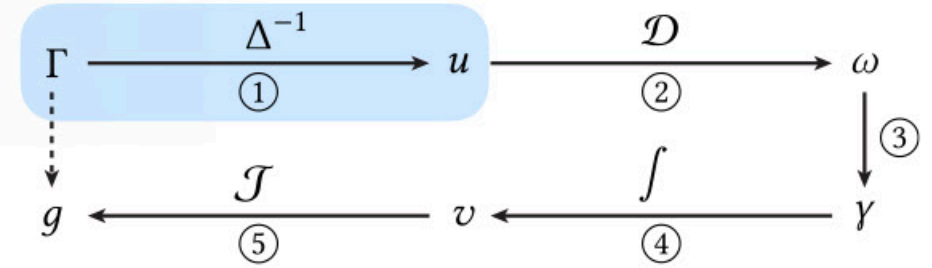
Reduced coordinates



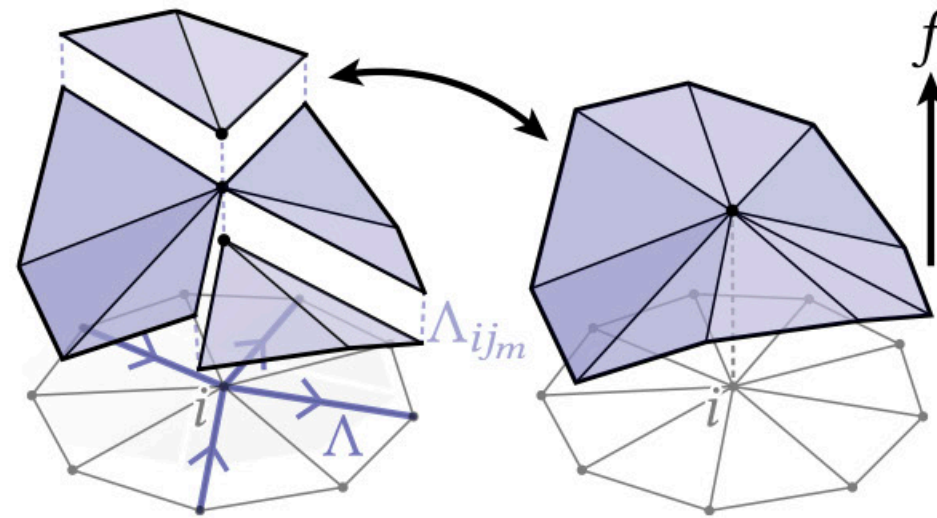
A jump harmonic function f is harmonic "up to jumps" Λ .



Reduced coordinates

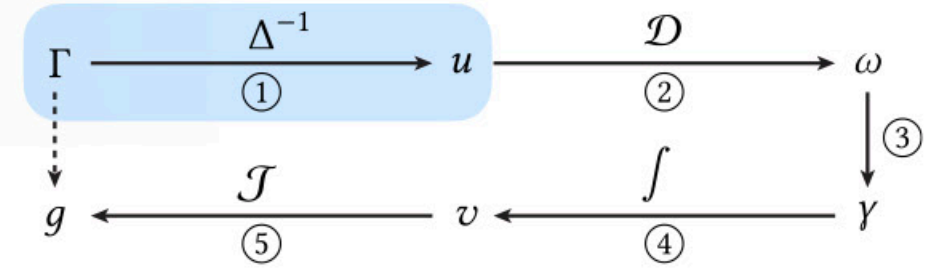


A jump harmonic function f is harmonic "up to jumps" Λ .

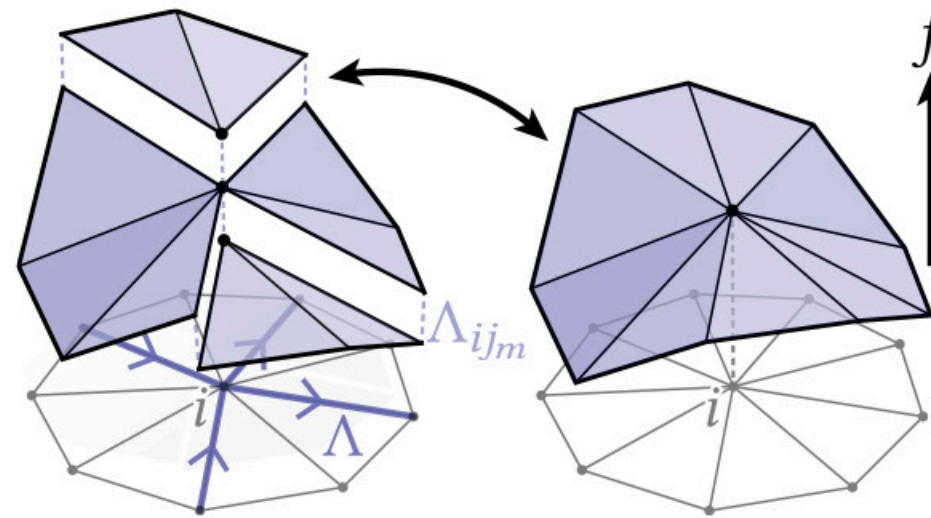


Value per corner \rightarrow
one value per vertex!

Reduced coordinates



A jump harmonic function f is harmonic "up to jumps" Λ .

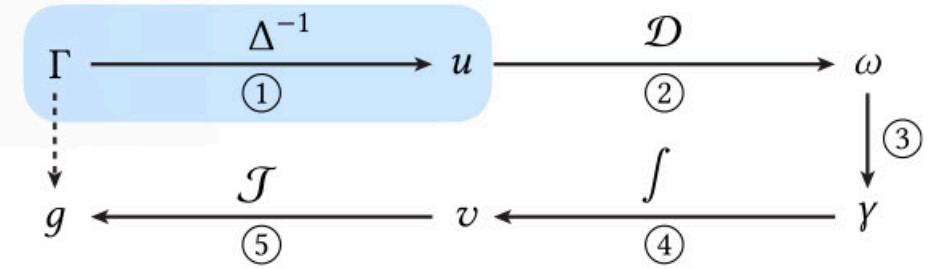


Value per corner →
one value per vertex!

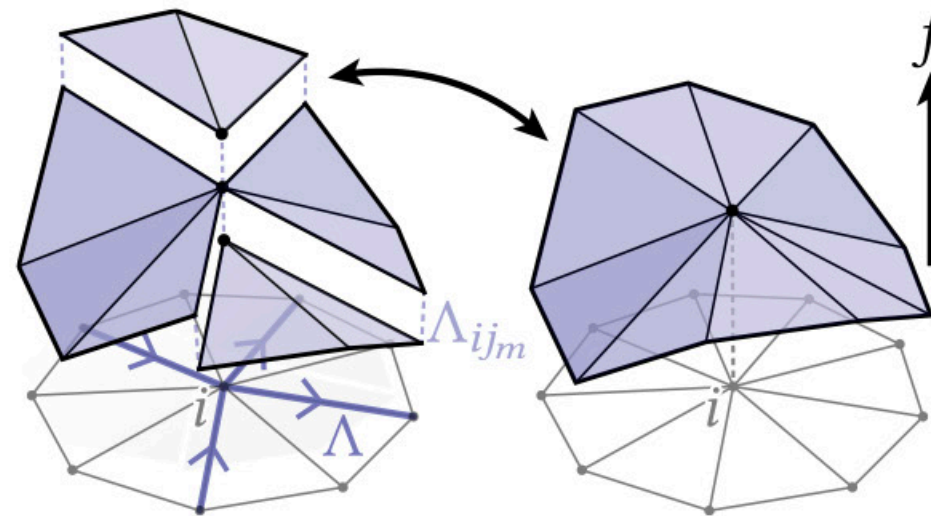
Apply change of variables and solve

$$L f_0 = b$$

Reduced coordinates



A jump harmonic function f is harmonic "up to jumps" Λ .



Value per corner →
one value per vertex!

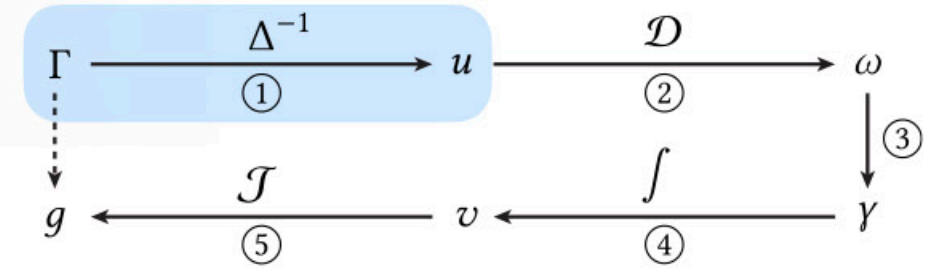
Apply change of variables and solve

values per vertex

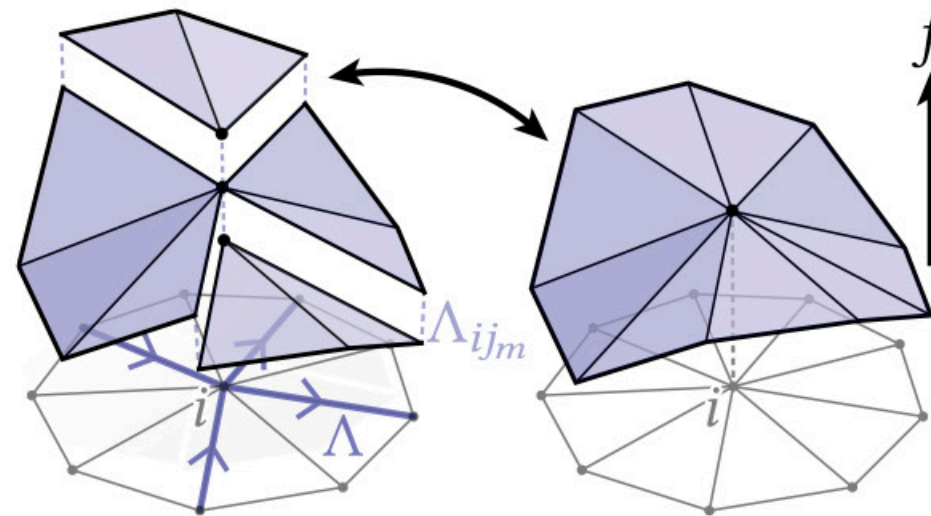
$$L\underline{f_0} = \underline{b}$$

constant vector encoding
per-corner jumps

Reduced coordinates



A jump harmonic function f is harmonic "up to jumps" Λ .



Value per corner →
one value per vertex!

*Designing Quadrangulations
with Discrete Harmonic Forms.*
Tong, Alliez, Cohen-Steiner,
Desbrun (2006)

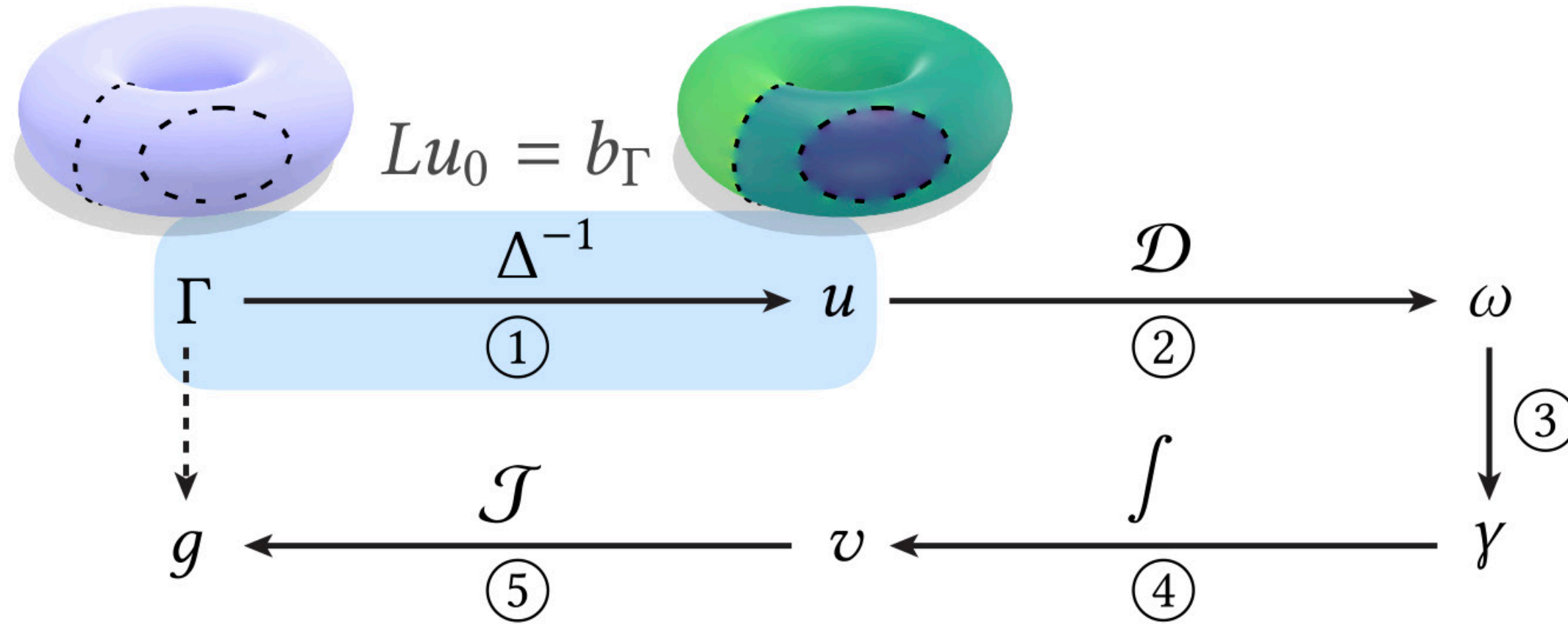
Apply change of variables and solve

values per vertex

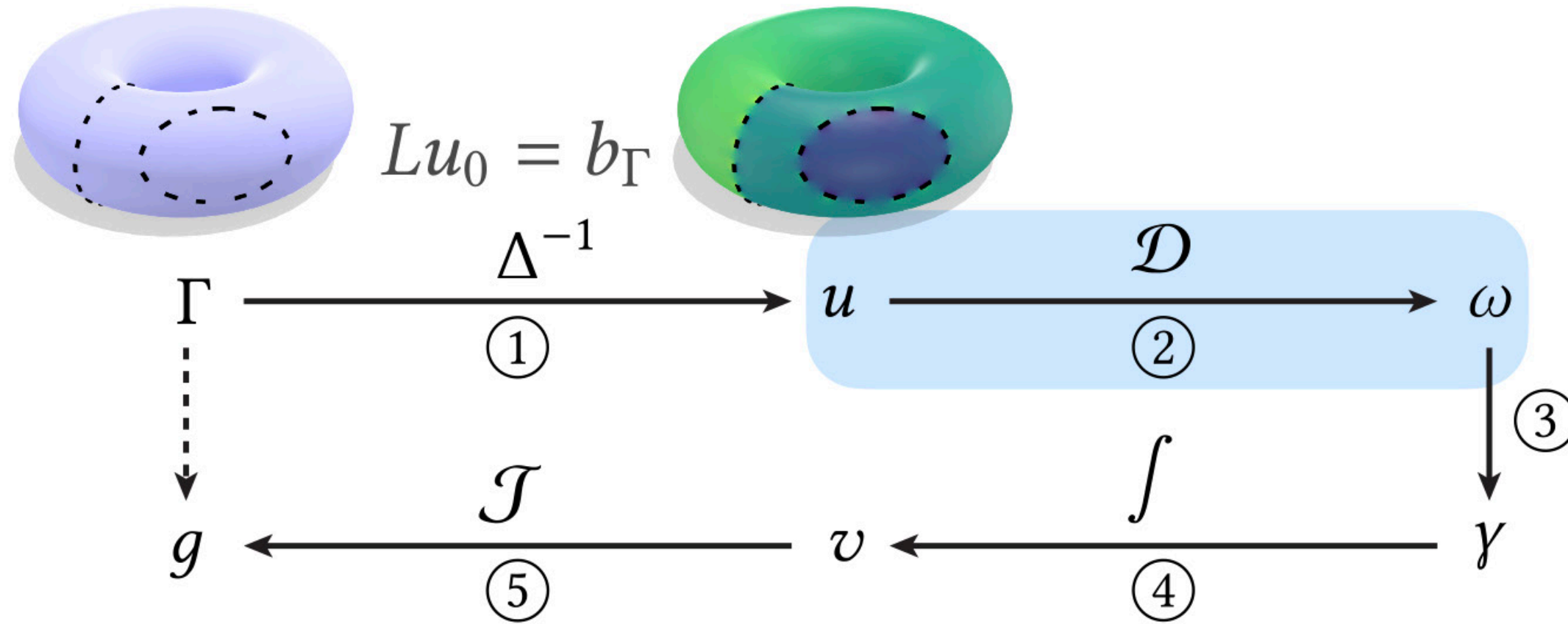
$$\underline{L} f_0 = \underline{b}$$

constant vector encoding
per-corner jumps

① SOLVE THE JUMP LAPLACE EQUATION FOR u

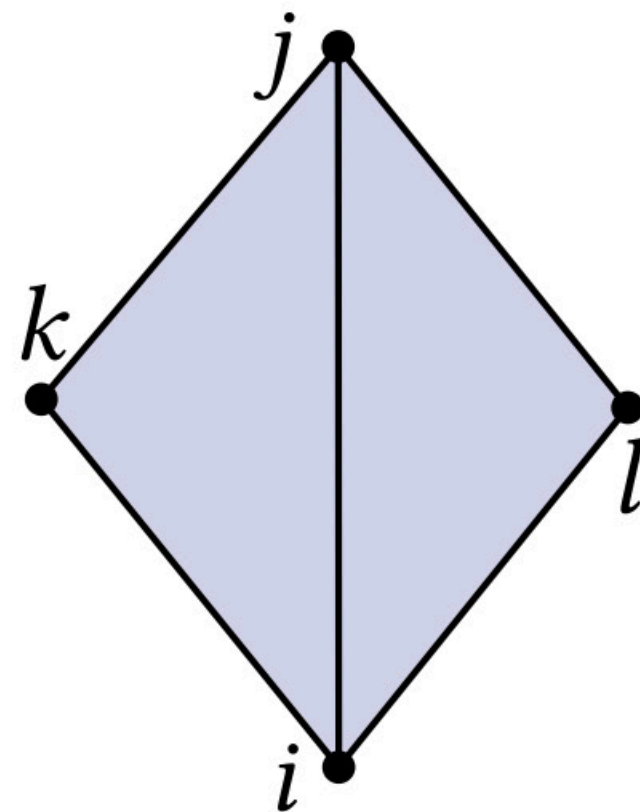
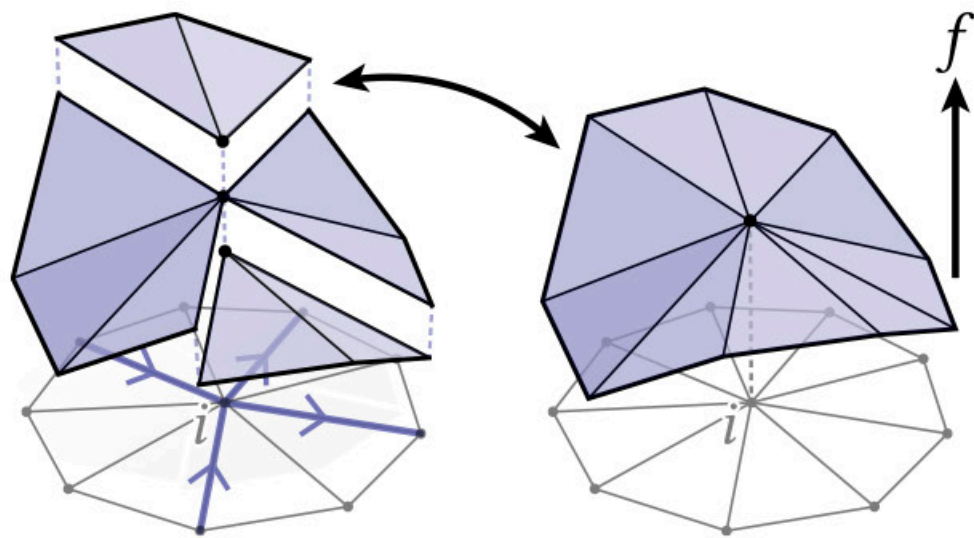


Jump harmonic functions \rightarrow Derivatives...



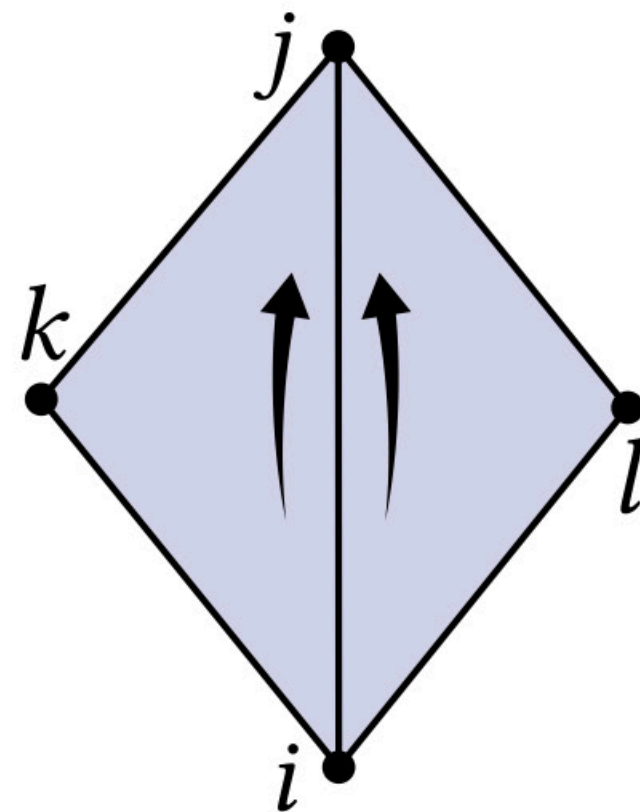
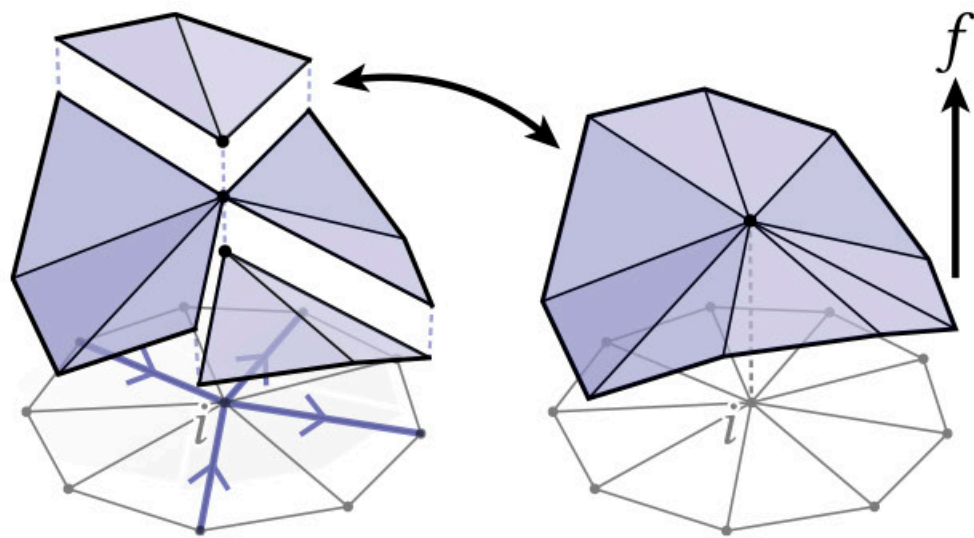
Differentiation

$$\begin{array}{ccccc}
 \Gamma & \xrightarrow[\textcircled{1}]{\Delta^{-1}} & u & \xrightarrow[\textcircled{2}]{\mathcal{D}} & \omega \\
 \vdots & & & & \downarrow \textcircled{3} \\
 g & \xleftarrow[\textcircled{5}]{\mathcal{J}} & v & \xleftarrow[\textcircled{4}]{f} & \gamma
 \end{array}$$

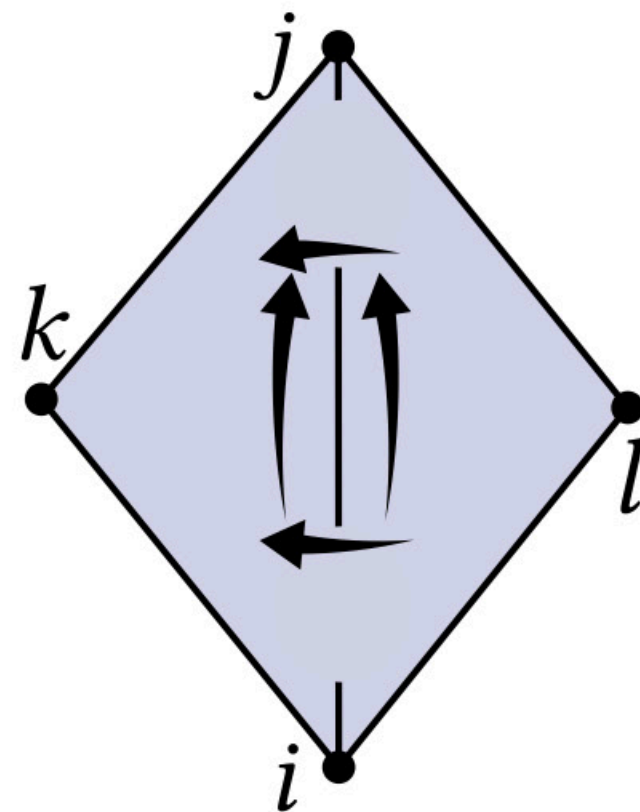
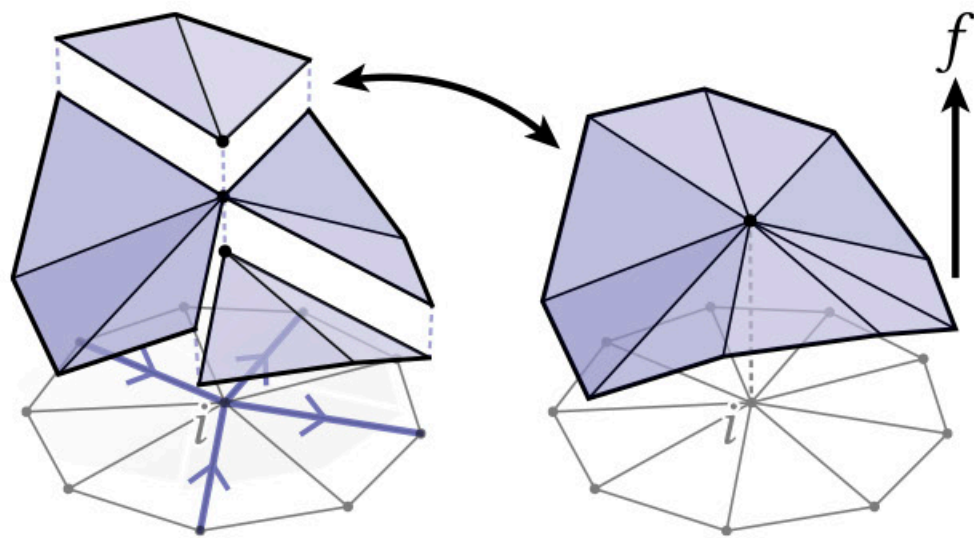
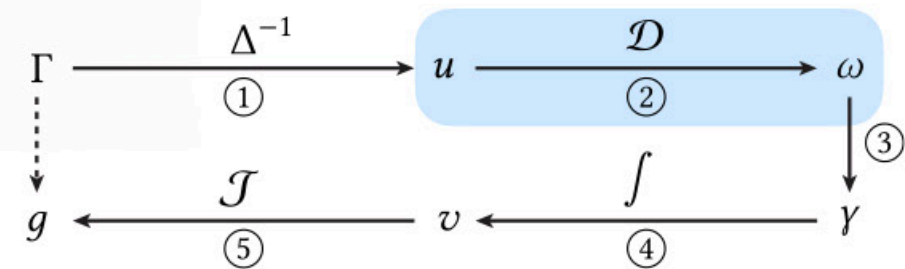


Differentiation

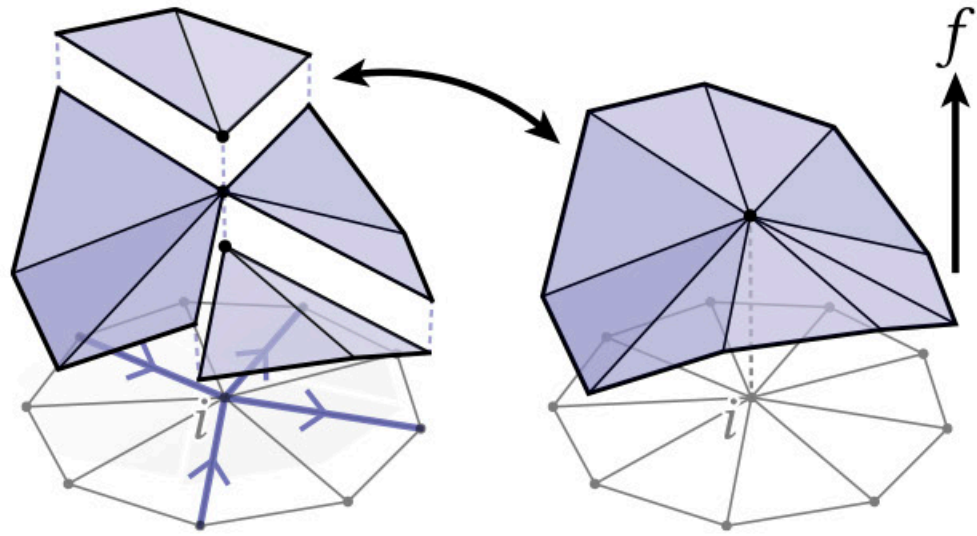
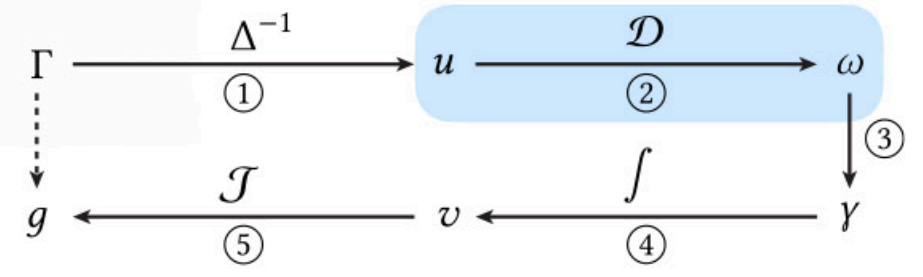
$$\begin{array}{ccccc}
 \Gamma & \xrightarrow[\textcircled{1}]{\Delta^{-1}} & u & \xrightarrow[\textcircled{2}]{\mathcal{D}} & \omega \\
 \vdots & & & & \downarrow \textcircled{3} \\
 g & \xleftarrow[\textcircled{5}]{\mathcal{J}} & v & \xleftarrow[\textcircled{4}]{f} & \gamma
 \end{array}$$



Differentiation

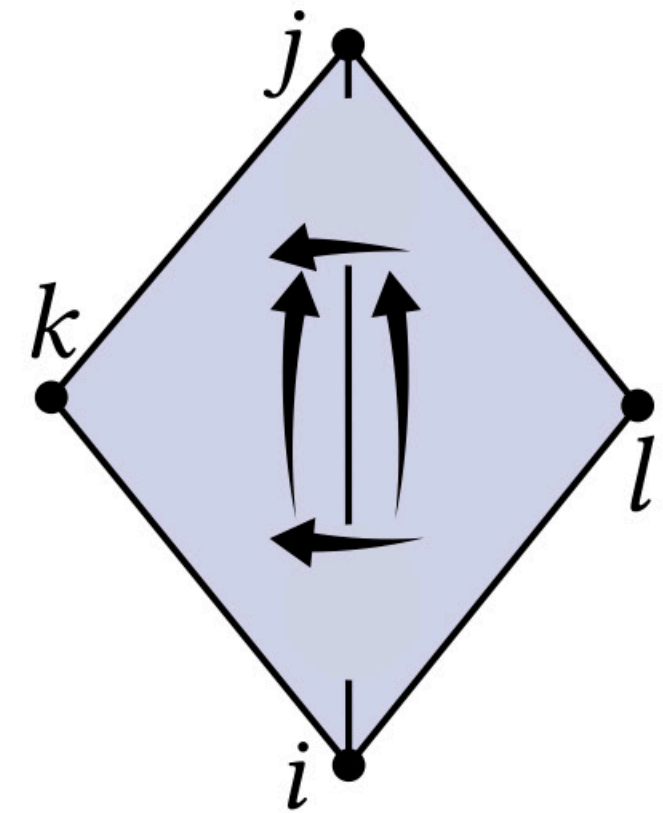


Differentiation

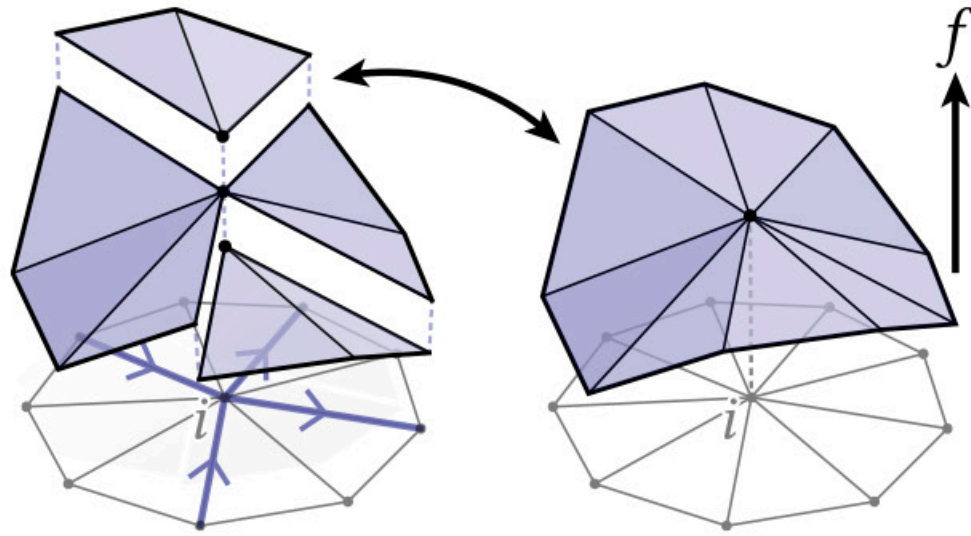
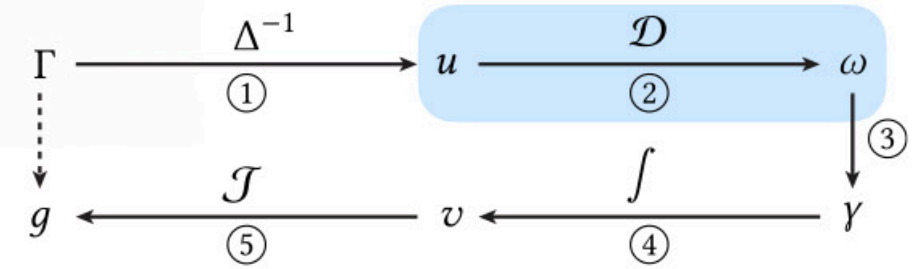


Jump compatibility condition:

$$f_i^{jk} - f_i^{lj} = f_j^{ki} - f_j^{il}$$



Differentiation



Jump compatibility condition:

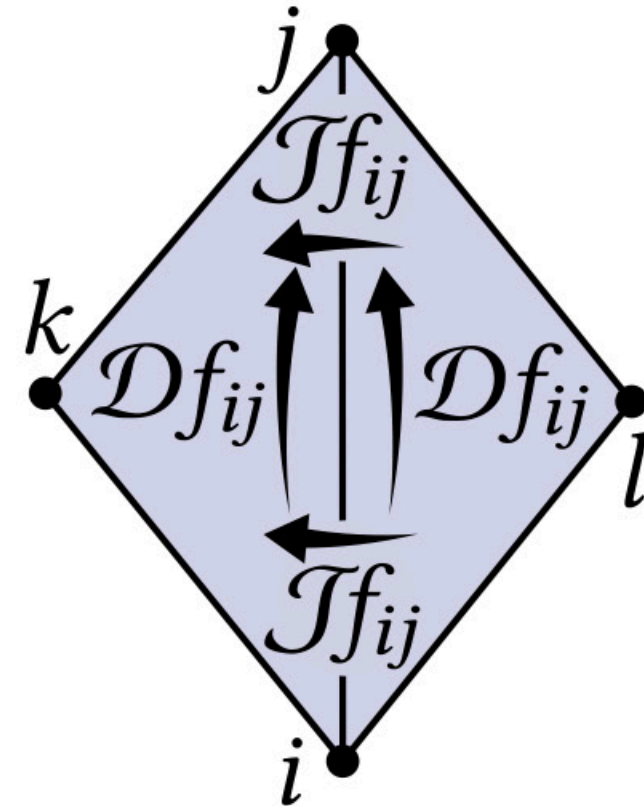
$$f_i^{jk} - f_i^{lj} = f_j^{ki} - f_j^{il}$$

Darboux derivative:

$$(\mathcal{D}f)_{ij} := f_j^{ki} - f_i^{jk}$$

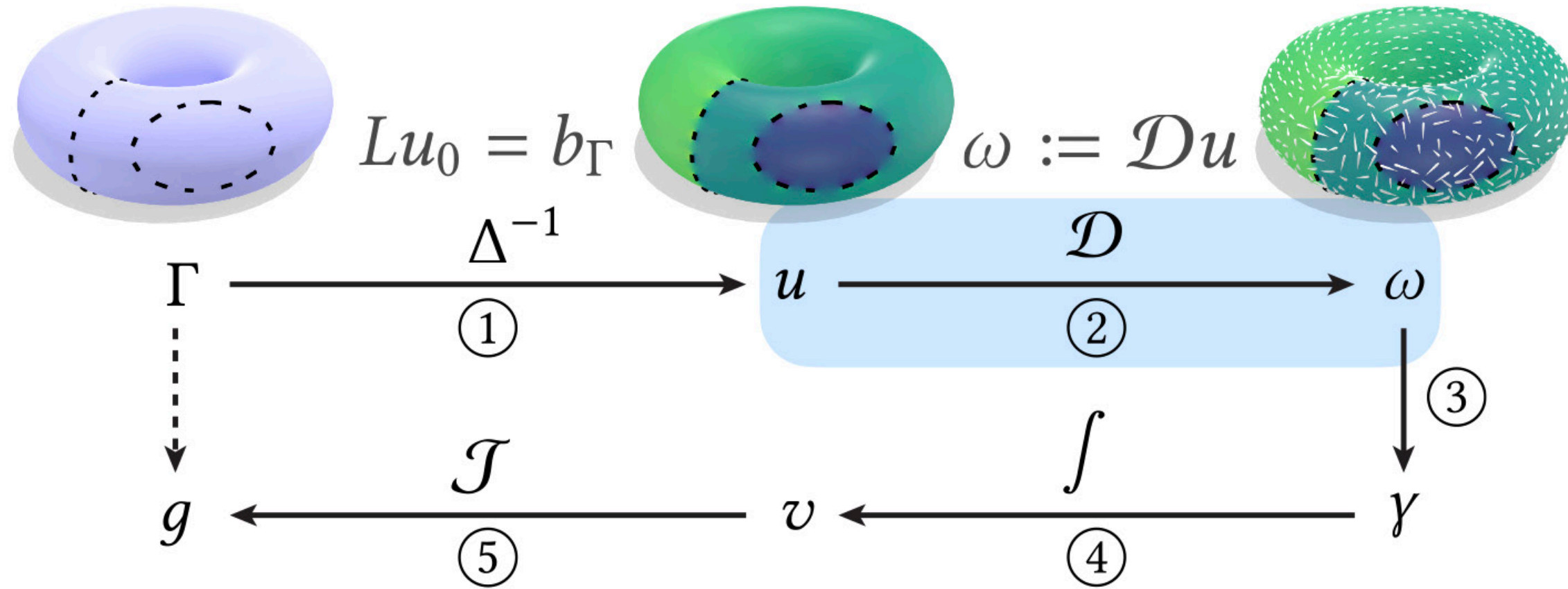
Jump derivative:

$$(\mathcal{J}f)_{ij} := f_i^{jk} - f_i^{lj}$$

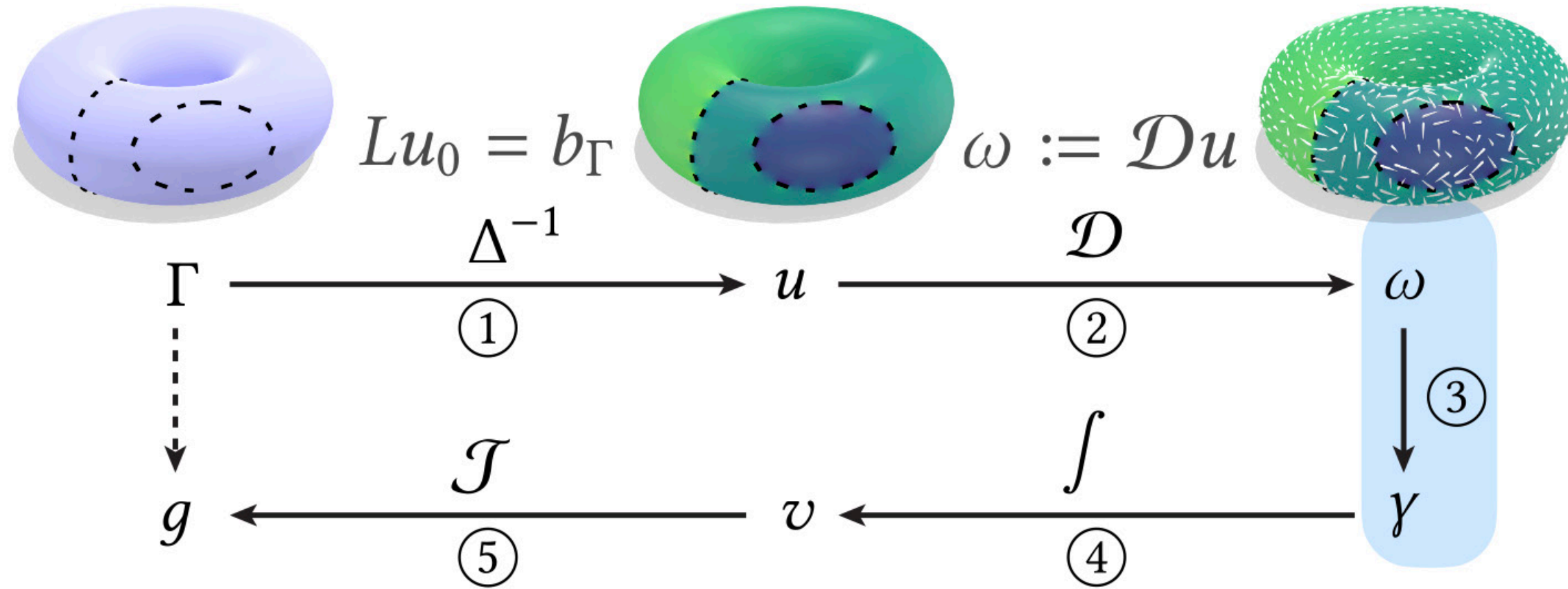


*subject to curve endpoints or nonmanifold edges

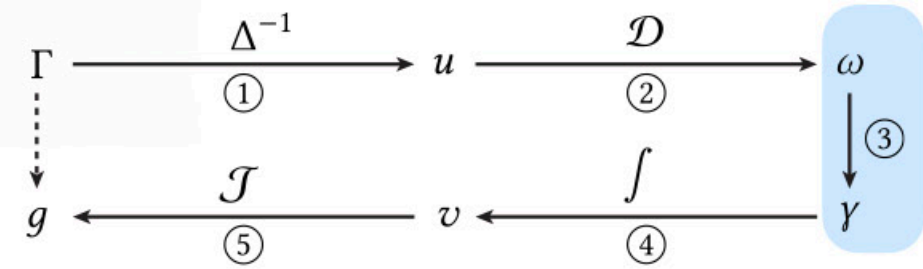
② DIFFERENTIATE u



Derivative decomposition...

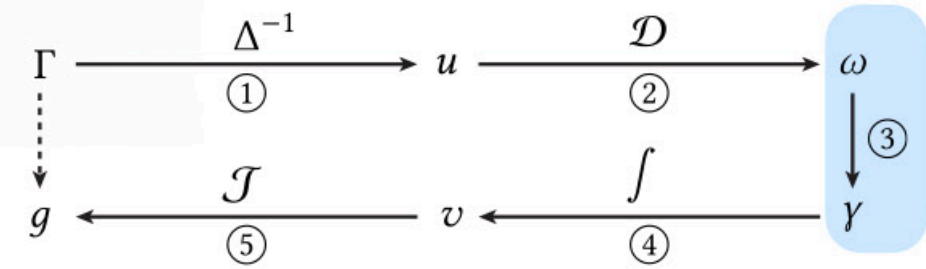


Hodge decomposition



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

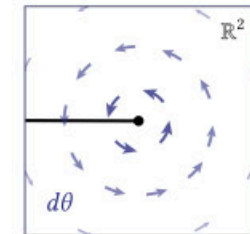
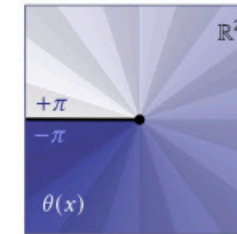
Hodge decomposition



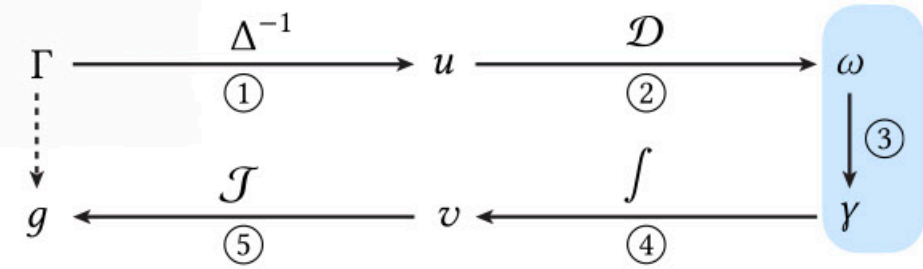
$$\omega = \cancel{d\alpha} + \delta\beta + \gamma$$

$\nearrow 0$
 \nearrow

Lemma, Appendix A

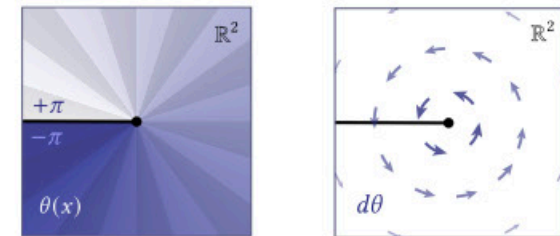


Hodge decomposition



$$\omega = \cancel{d\alpha} + \delta\beta + \gamma$$

Lemma, Appendix A



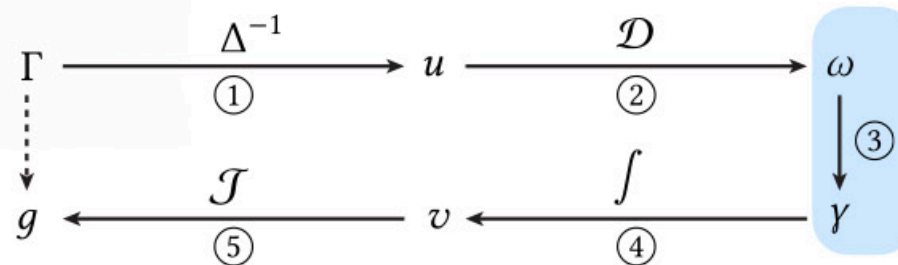
Solve Poisson equation:

$$\Delta_2 \beta = d_1 \omega$$

$$\Delta_2 := d_1 *_{1}^{-1} d_1^T *_{2}$$

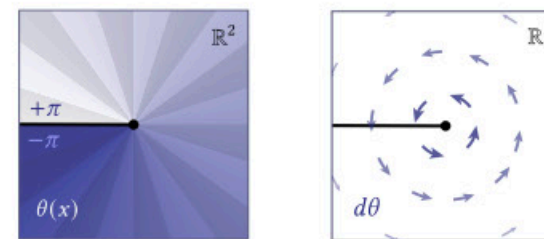
Discrete differential forms for computational modeling.
Desbrun, Kanso, Tong (2005)

Hodge decomposition



$$\omega = \cancel{d\alpha} + \delta\beta + \gamma$$

Lemma, Appendix A



Solve Poisson equation:

$$\Delta_2 \beta = d_1 \omega$$

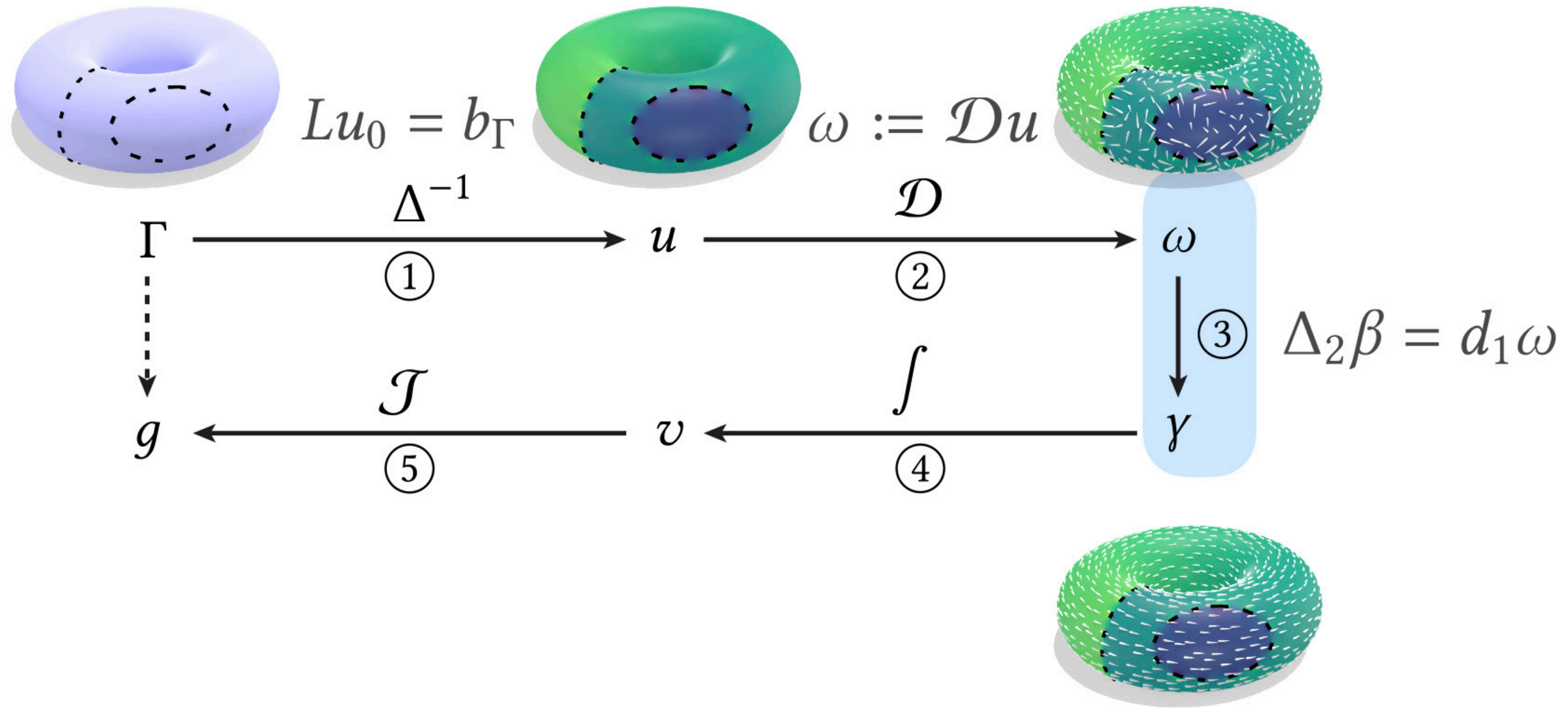
$$\Delta_2 := d_1 *_{1}^{-1} d_1^T *_{2}$$

Get harmonic component:

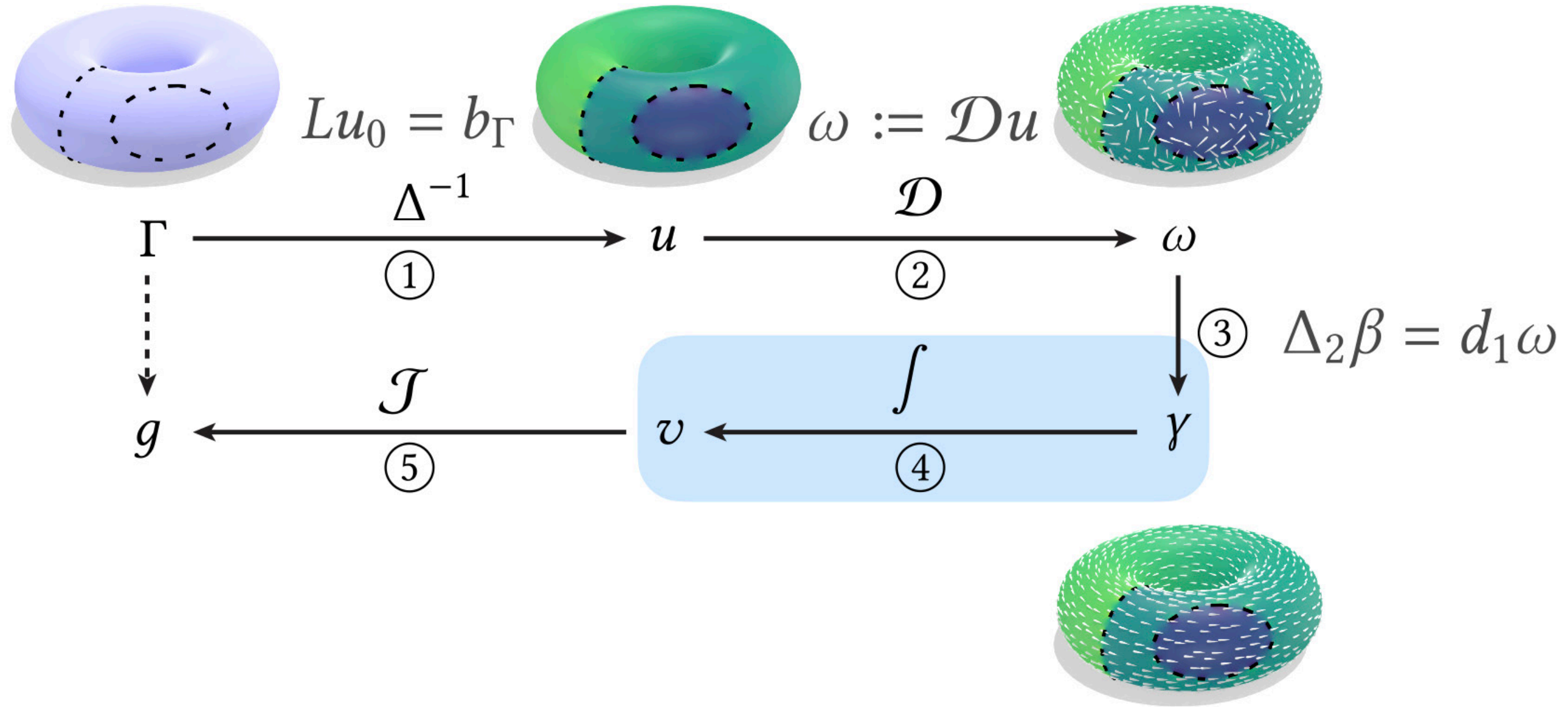
$$\gamma \leftarrow \omega - \delta\beta$$

Discrete differential forms for computational modeling.
Desbrun, Kanso, Tong (2005)

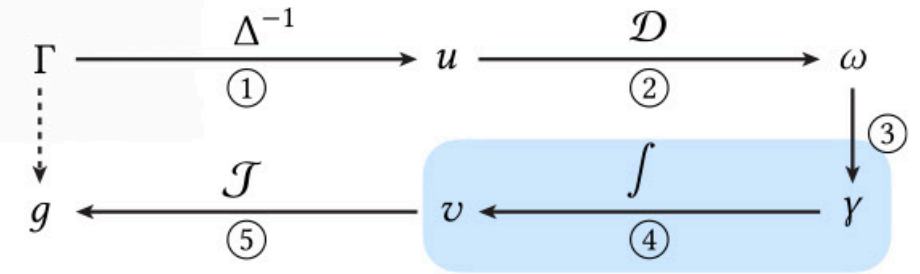
③ HODGE DECOMPOSE ω



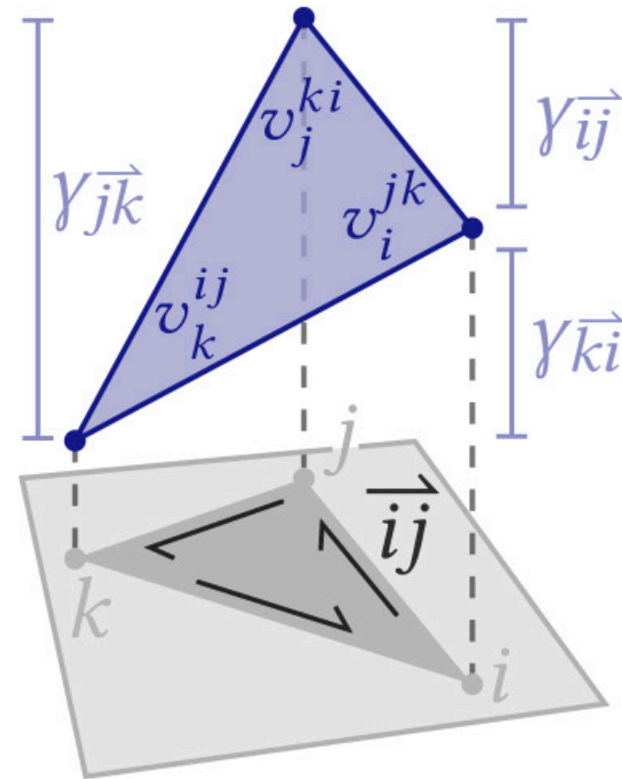
1-forms \rightarrow Jump harmonic functions...



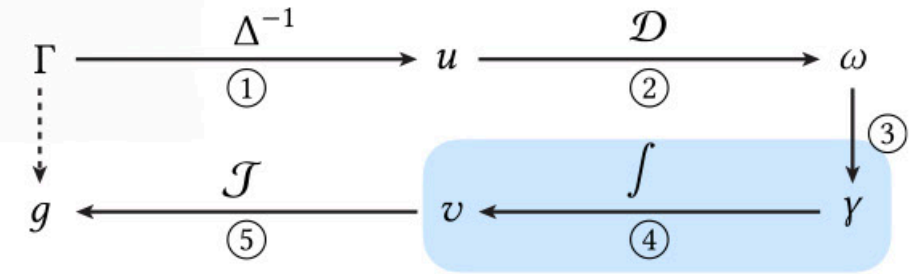
Integrating γ with jumps



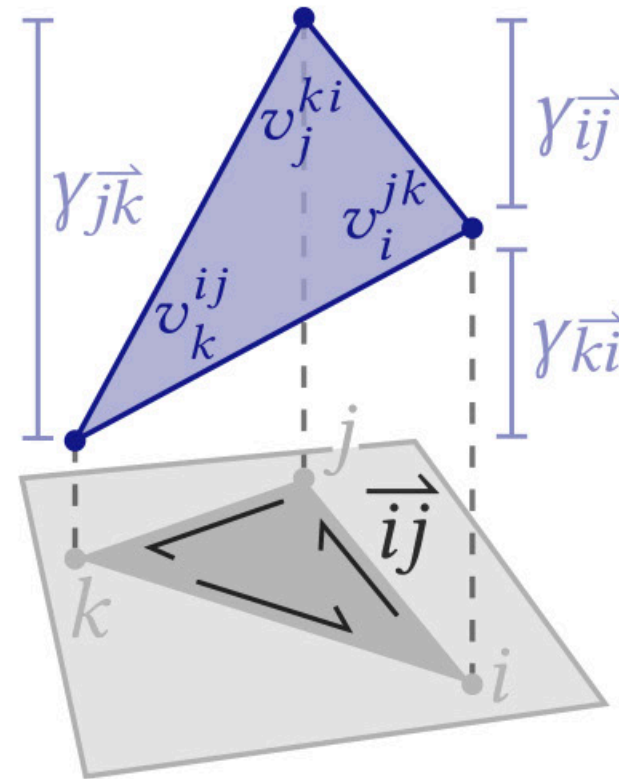
First locally integrate γ in each triangle.



Integrating γ with jumps

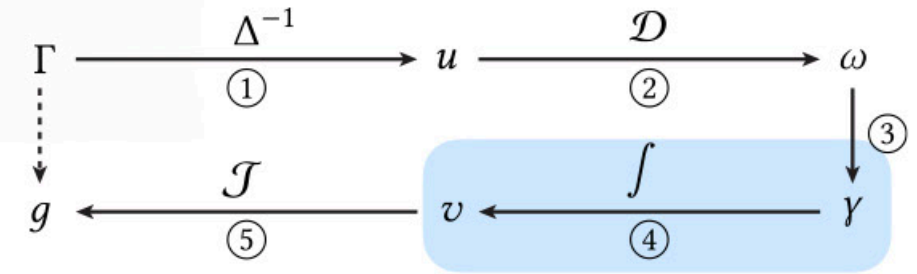


First locally integrate γ in each triangle.

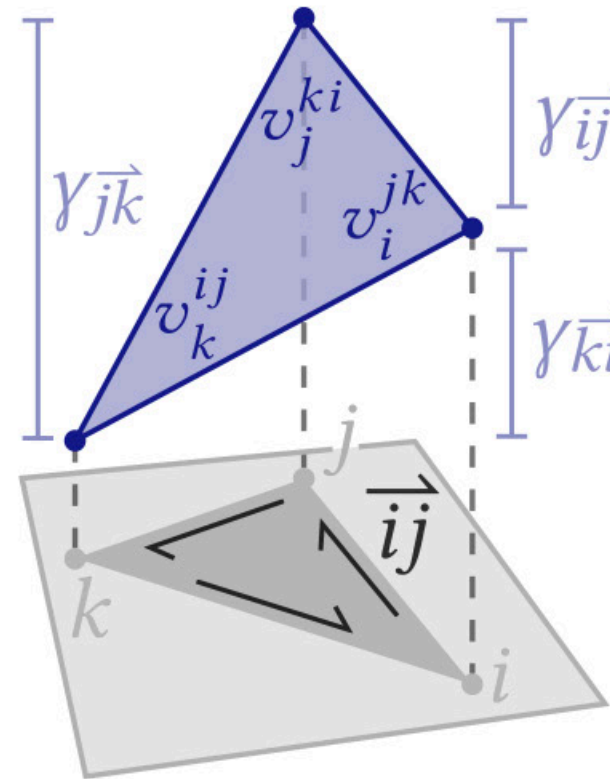


$$\overset{\circ}{v}_i^{jk} := 0$$

Integrating γ with jumps



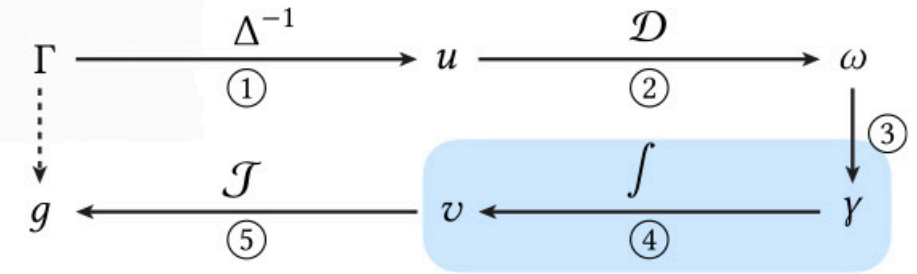
First locally integrate γ in each triangle.



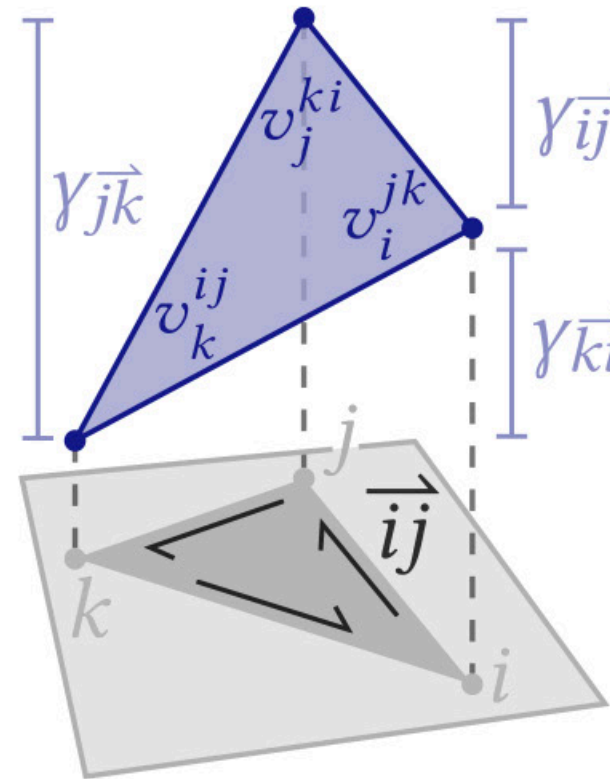
$$\overset{\circ}{v}_i^{jk} := 0$$

$$\overset{\circ}{v}_j^{ki} := \gamma_{ij}$$

Integrating γ with jumps



First locally integrate γ in each triangle.

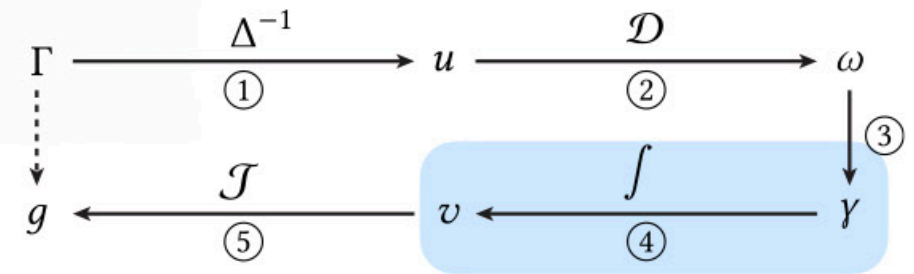


$$\mathring{v}_i^{jk} := 0$$

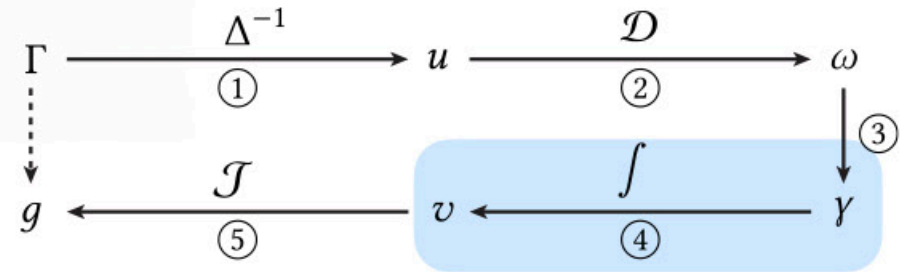
$$\mathring{v}_j^{ki} := \gamma_{ij}$$

$$\mathring{v}_k^{ij} := \gamma_{ij} + \gamma_{jk}$$

Computing the residual function



Computing the residual function

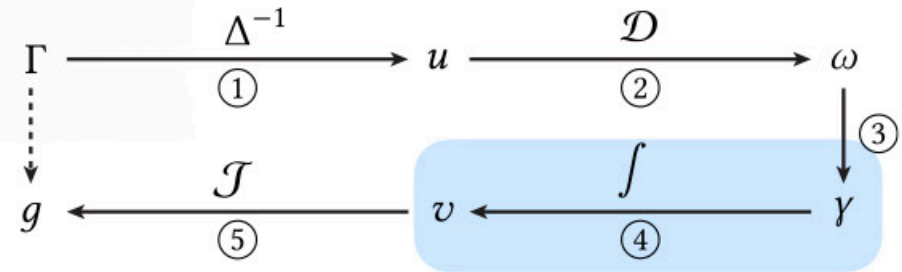


Applying a change of variable, optimize for $|F|$ per-face shifts:

$$\begin{aligned}
 & \min_{\sigma \in \mathbb{R}^{|F|}} \sum_{ij \in \Gamma \cap E_{\text{int}}} \ell_{ij} |(\sigma_{ijk} - \sigma_{jil}) - s_{ij}| + \frac{1}{\varepsilon} \sum_{ij \in E_{\text{int}} \setminus \Gamma} \ell_{ij} |(\sigma_{ijk} - \sigma_{jil}) - s_{ij}| \\
 & \text{s.t. } 0 \leq \frac{(\sigma_{ijk} - \sigma_{jil}) - s_{ij}}{\Gamma_{ij}} \leq 1, \quad \forall ij \in \Gamma.
 \end{aligned}$$

linear program

Computing the residual function



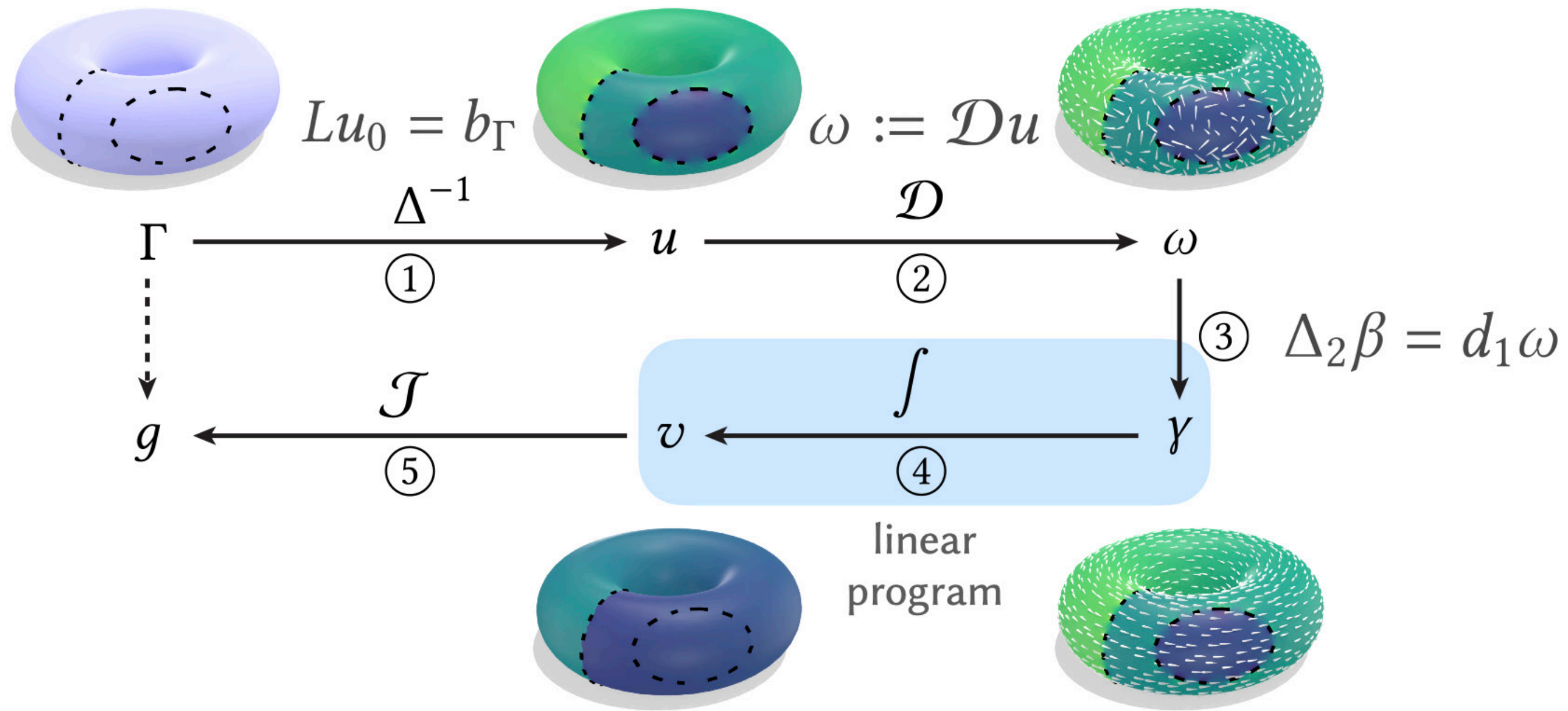
Applying a change of variable, optimize for $|F|$ per-face shifts:

$$\begin{aligned}
 & \min_{\sigma \in \mathbb{R}^{|F|}} \sum_{ij \in \Gamma \cap E_{\text{int}}} \ell_{ij} |(\sigma_{ijk} - \sigma_{jil}) - s_{ij}| + \frac{1}{\varepsilon} \sum_{ij \in E_{\text{int}} \setminus \Gamma} \ell_{ij} |(\sigma_{ijk} - \sigma_{jil}) - s_{ij}| \\
 & \text{s.t. } 0 \leq \frac{(\sigma_{ijk} - \sigma_{jil}) - s_{ij}}{\Gamma_{ij}} \leq 1, \quad \forall ij \in \Gamma.
 \end{aligned}$$

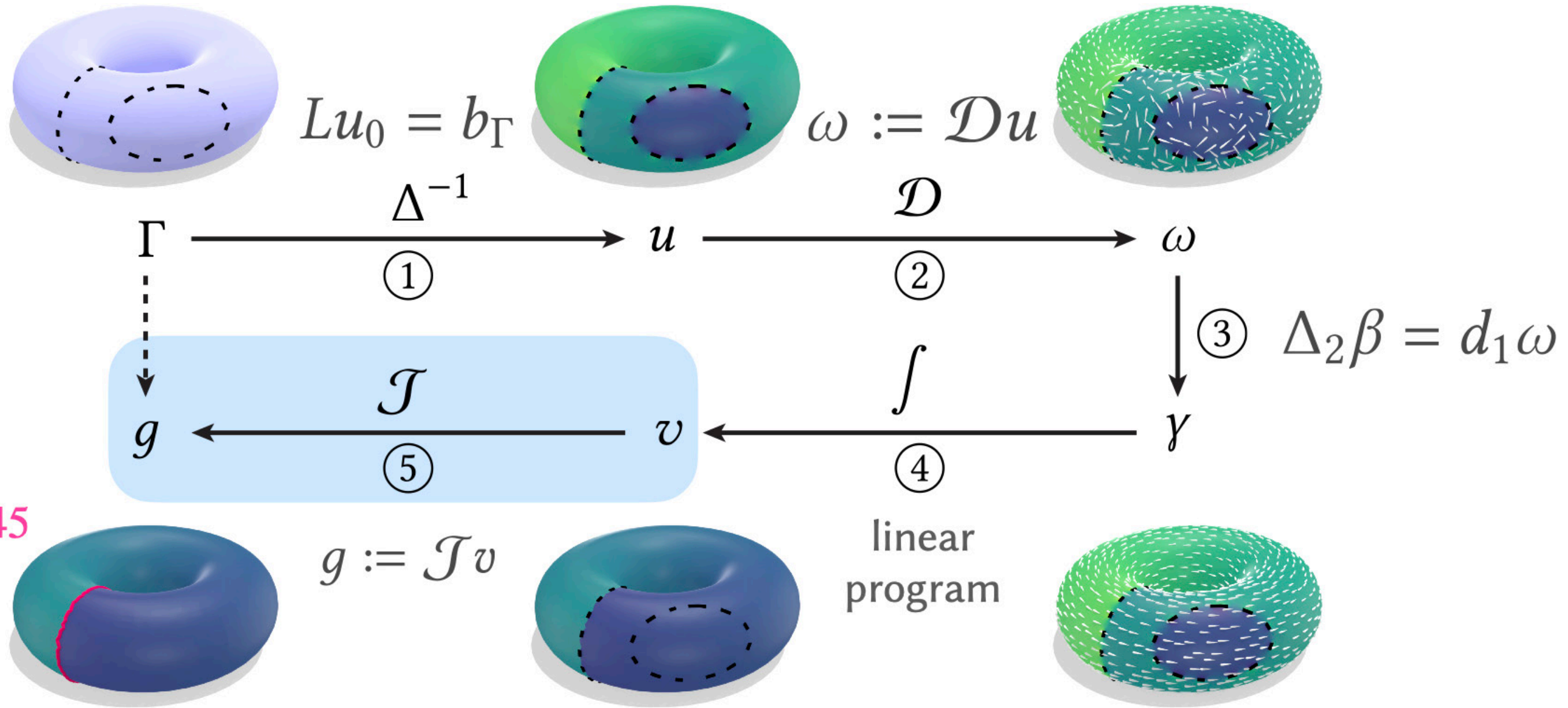
linear program

Afterwards, recover solution v .

④ COMPUTE THE RESIDUAL FUNCTION



⑤ DECOMPOSE CURVE

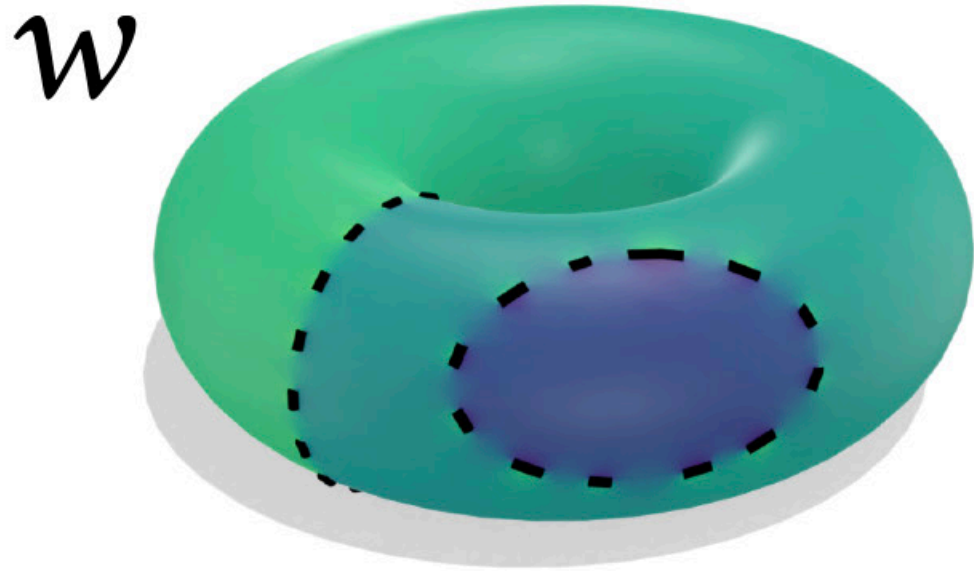


Winding number function

Solve jump Laplace equation with jumps $\Gamma - \mathcal{J}v$ to obtain w .

Winding number function

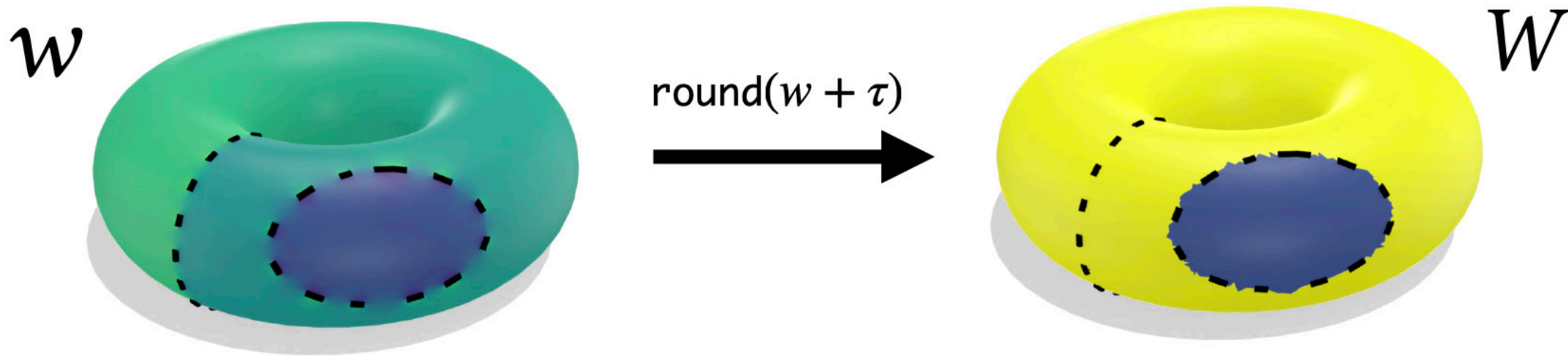
Solve jump Laplace equation with jumps $\Gamma - \mathcal{J}v$ to obtain w .



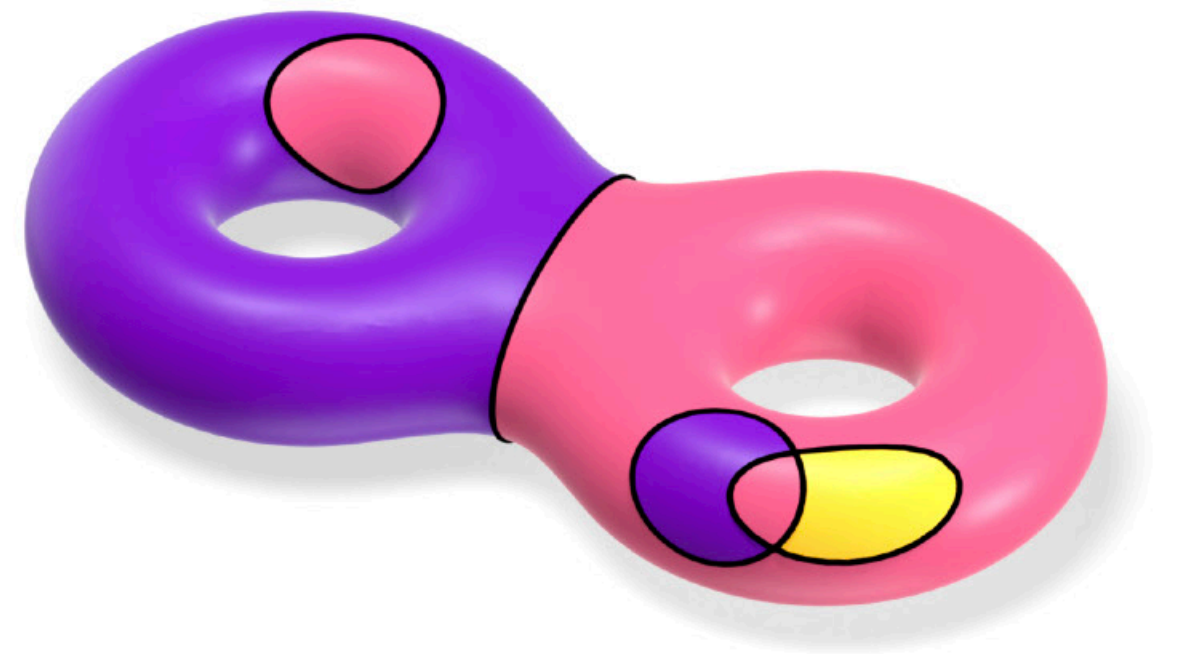
Winding number function

Solve jump Laplace equation with jumps $\Gamma - \mathcal{J}v$ to obtain w .

We compute a global shift τ such that $w + \tau$ is integer along Γ .

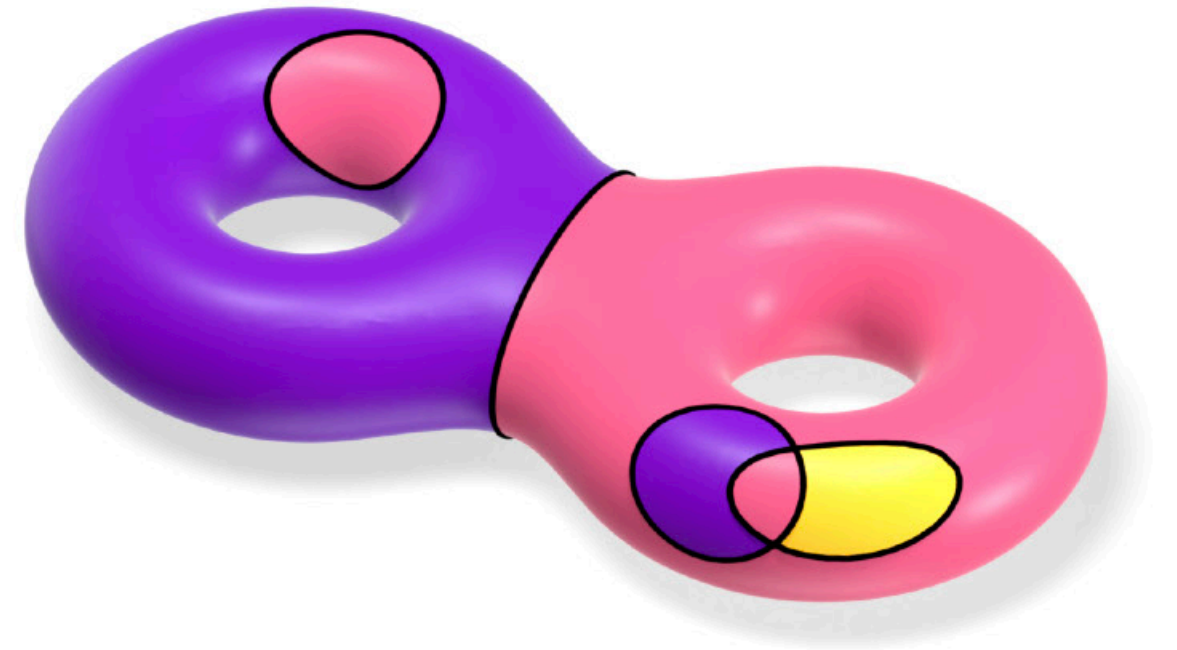


Summary



Summary

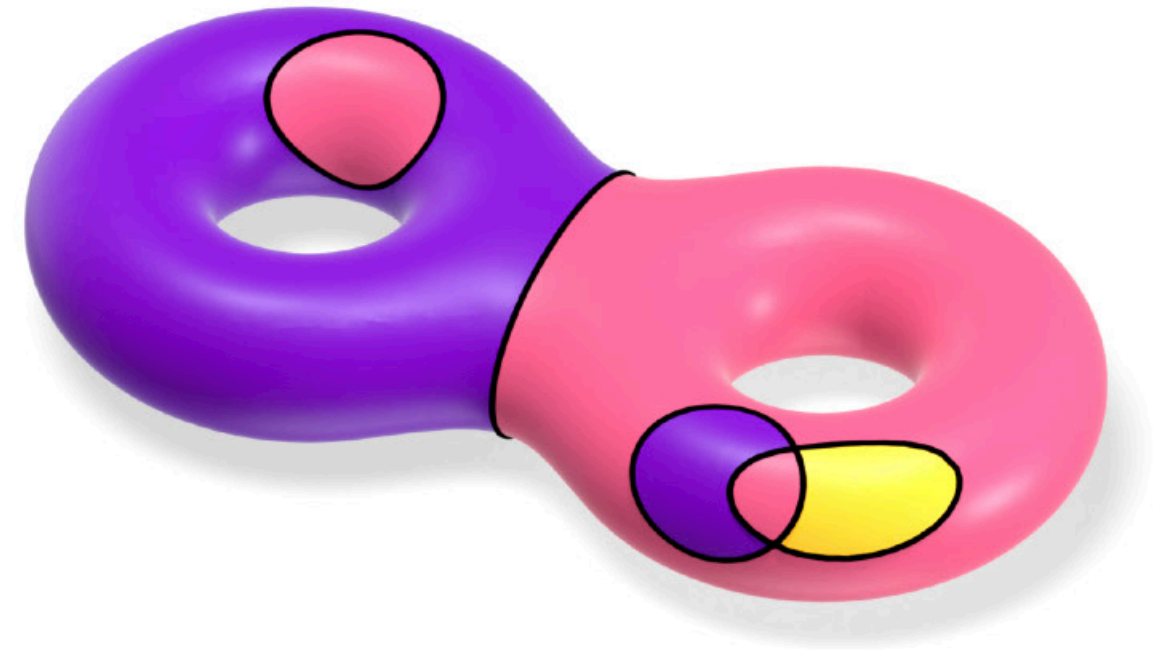
It's difficult to reason about the homology class of *broken* curves.



Summary

It's difficult to reason about the homology class of *broken* curves.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

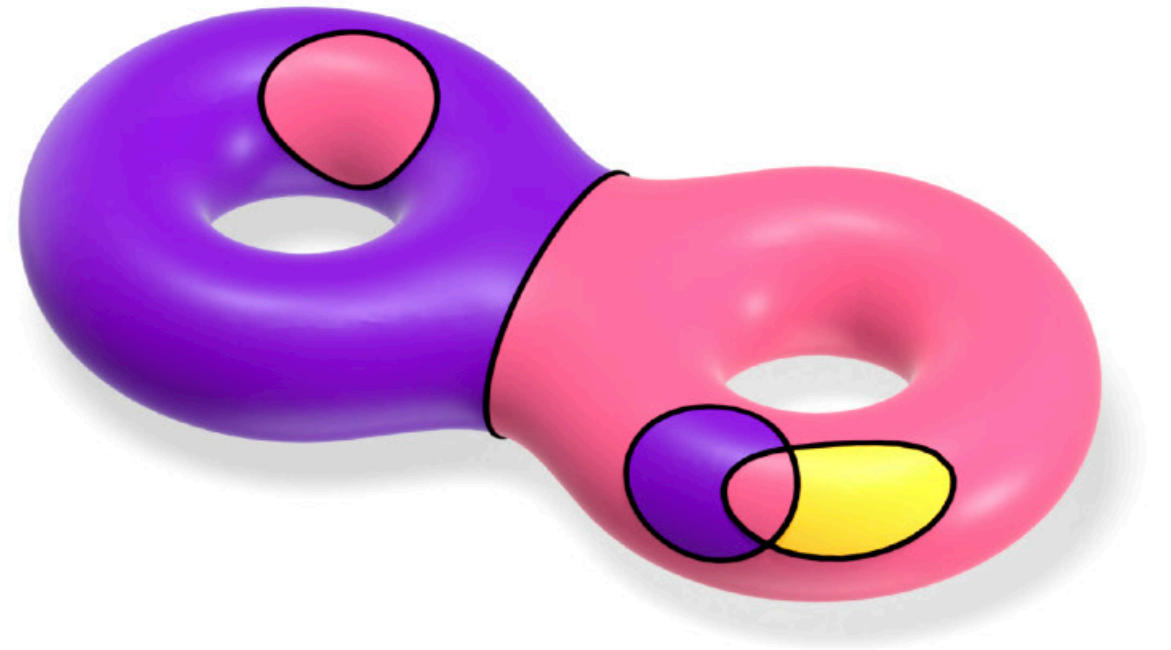


Summary

It's difficult to reason about the homology class of *broken* curves.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

We map from functions back to curves, yielding integer region labels & identification of nonbounding components.

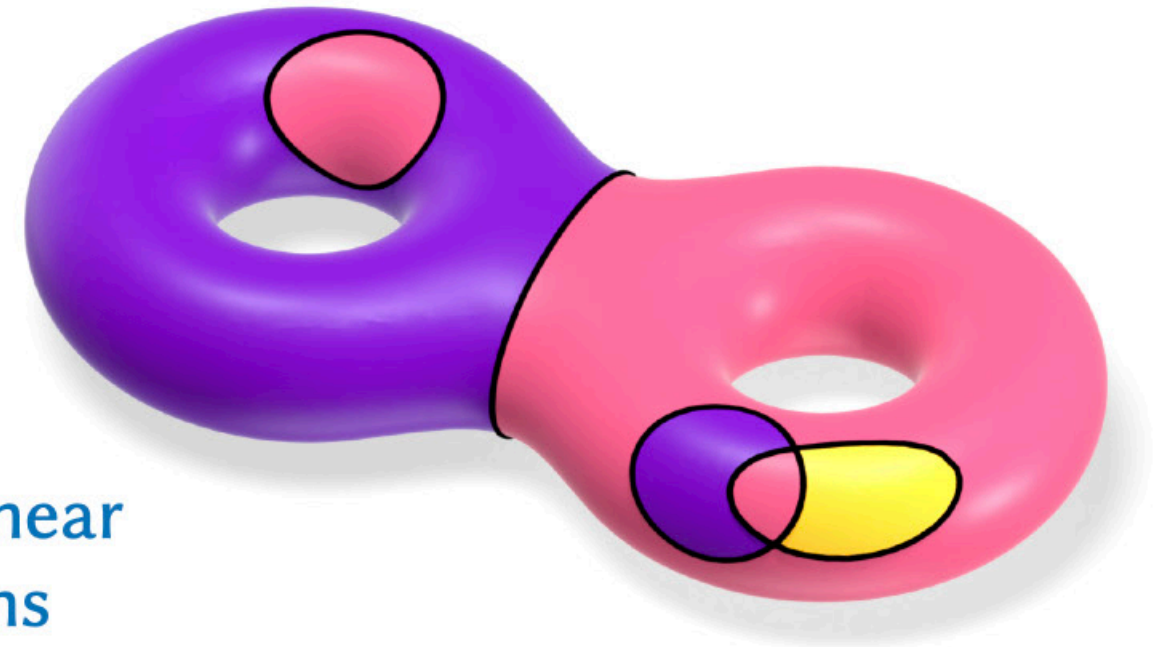


Summary

It's difficult to reason about the homology class of *broken* curves.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

sparse linear systems



We map from functions back to curves, yielding integer region labels & identification of nonbounding components.

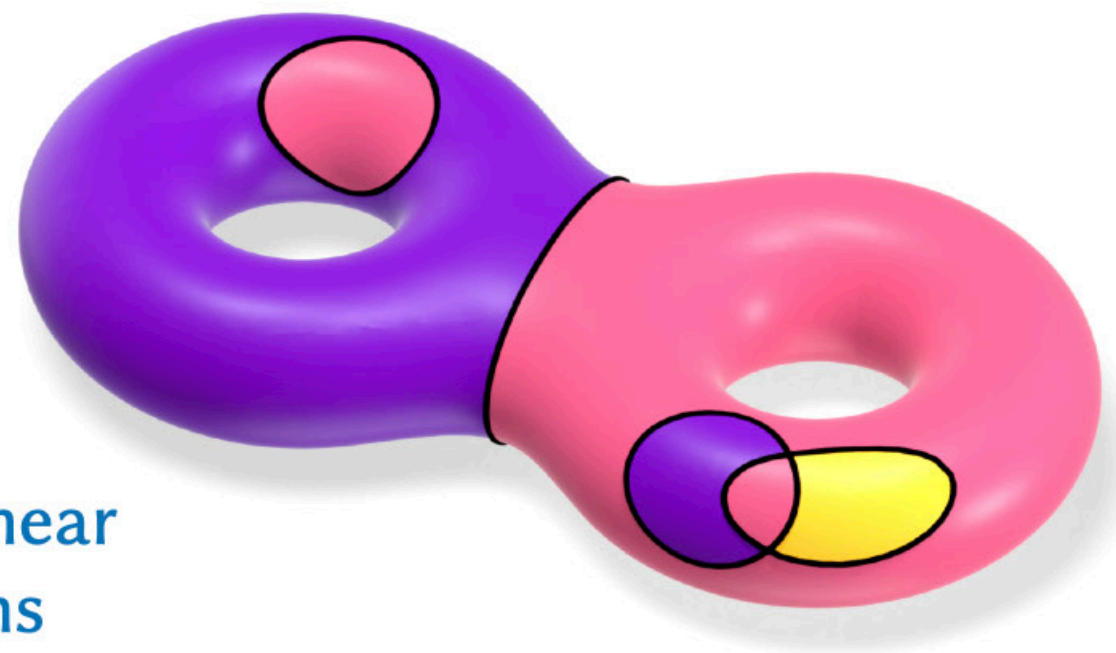
Summary

It's difficult to reason about the homology class of *broken* curves.

Instead of processing curves directly, we process functions *dual* to curves using *de Rham cohomology*.

We map from functions back to curves, yielding integer region labels & identification of nonbounding components.

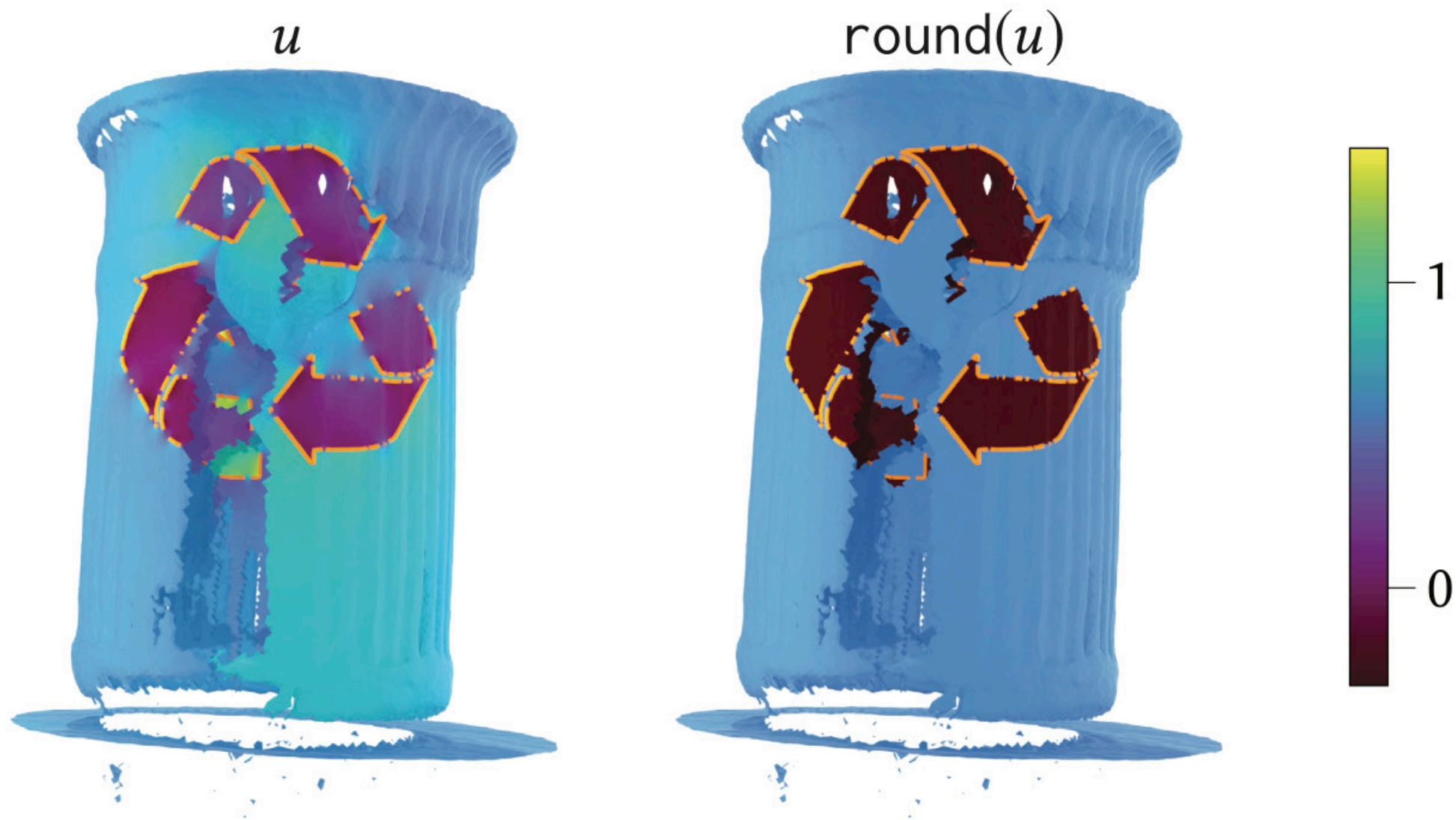
sparse linear
systems



sparse linear
program

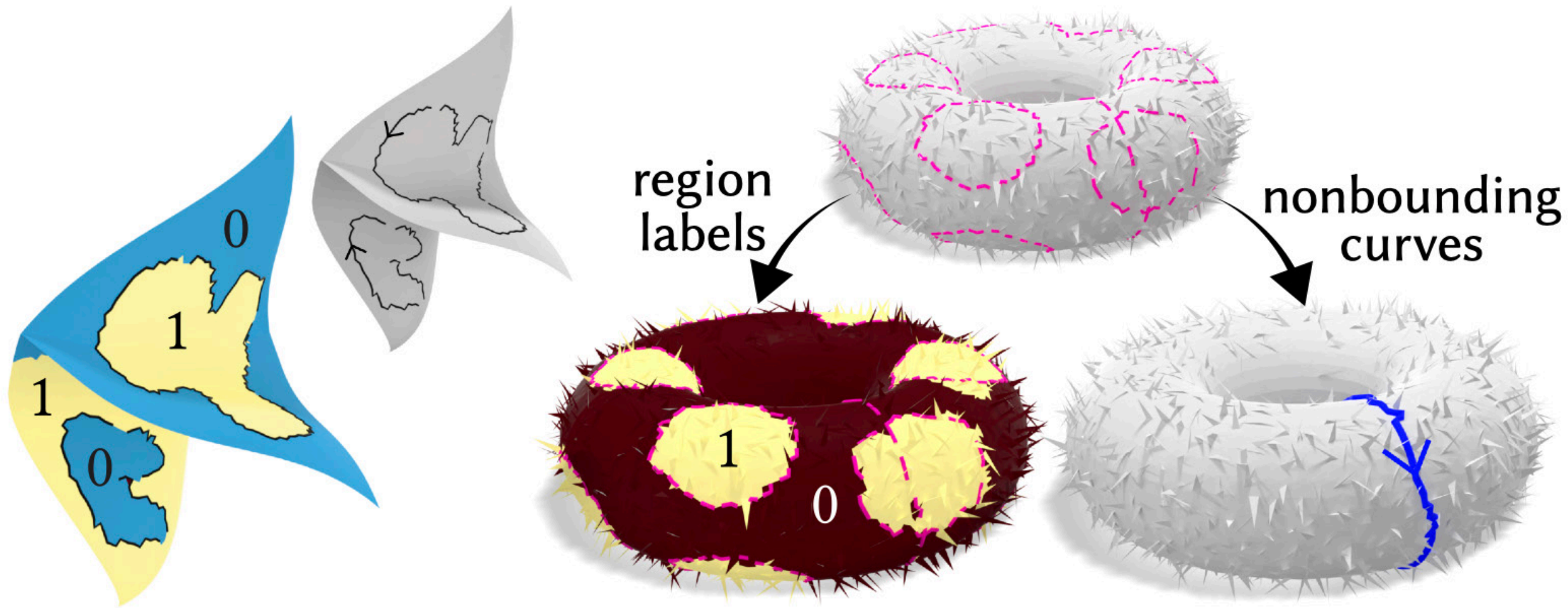
RESULTS

Robustness to defects in both Γ and M



holes, scanner noise

Robustness to defects in both Γ and M

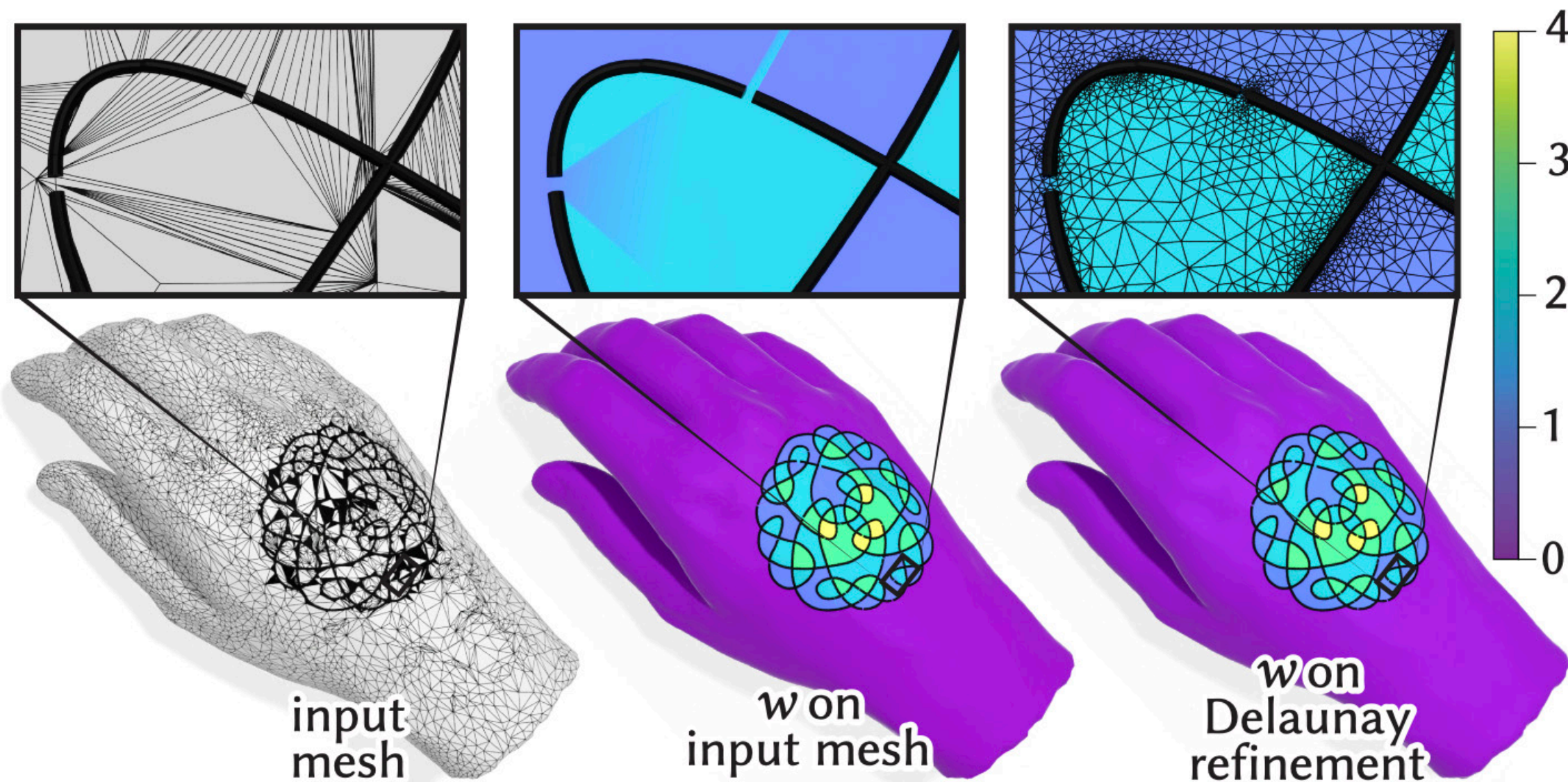


highly non-manifold surfaces

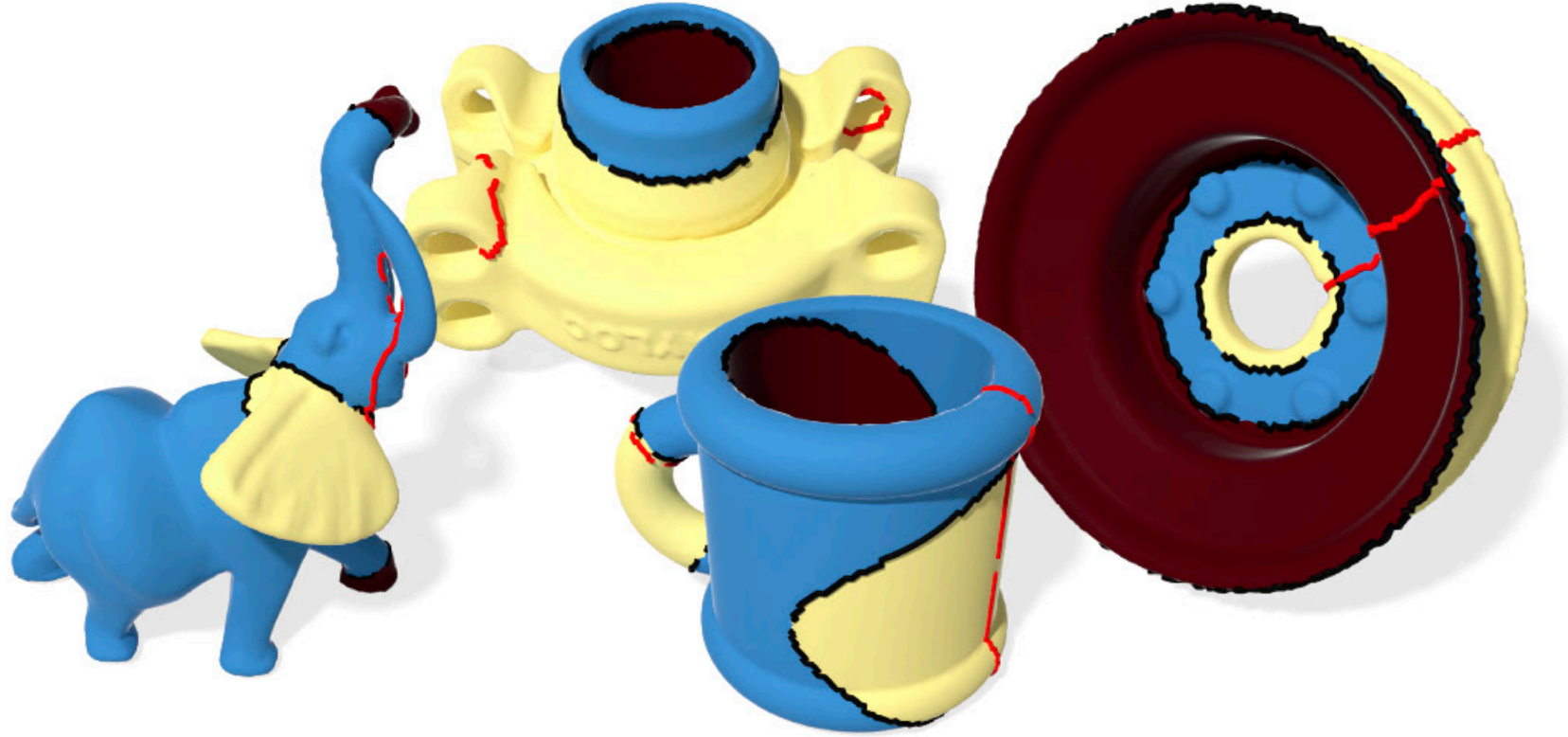
Intrinsic retriangulation

Navigating Intrinsic Triangulations. Sharp, Soliman, Crane (2019)

Integer Coordinates for Intrinsic Geometry Processing. Gillespie, Sharp, Crane (2021)

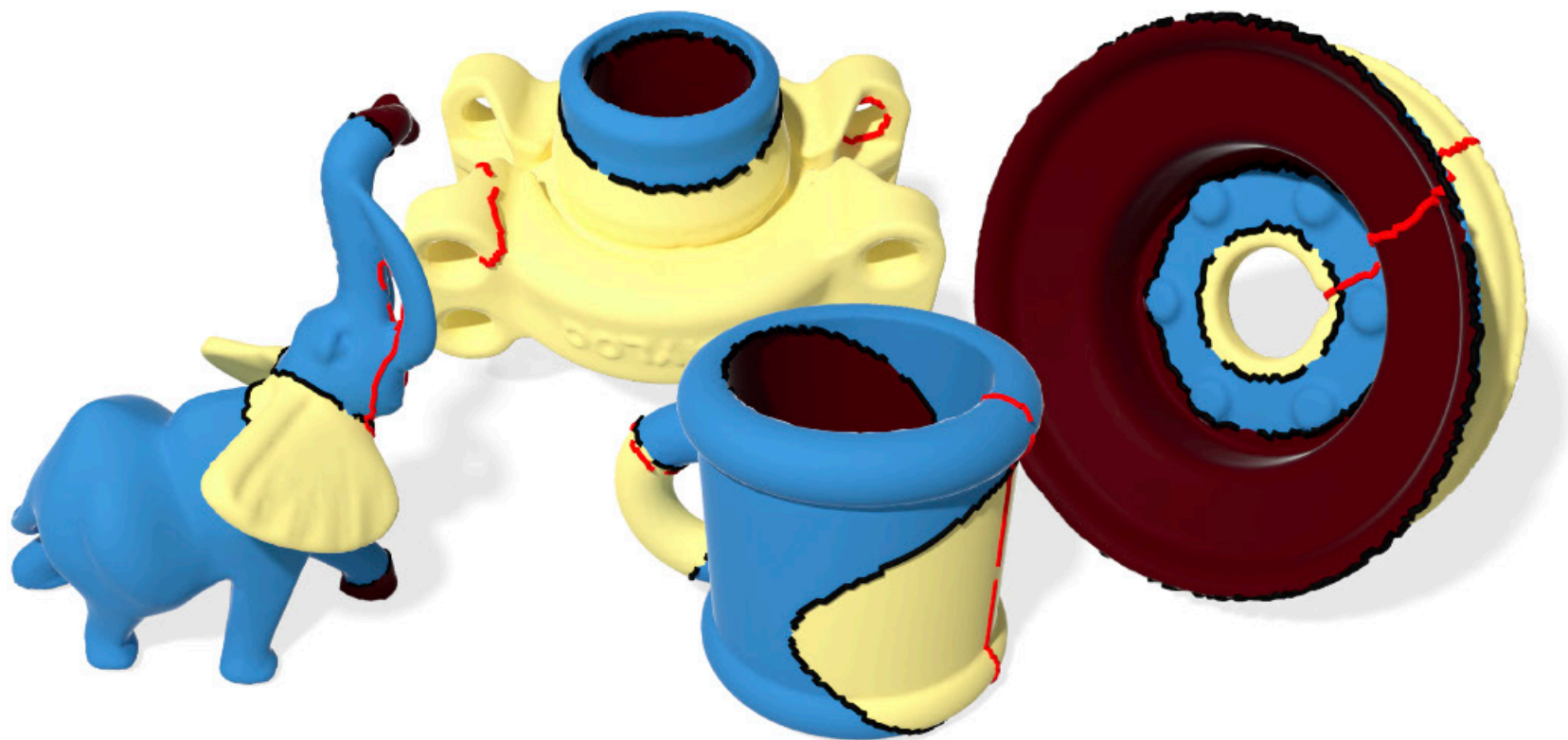


Evaluation



Benchmark setup

Evaluation

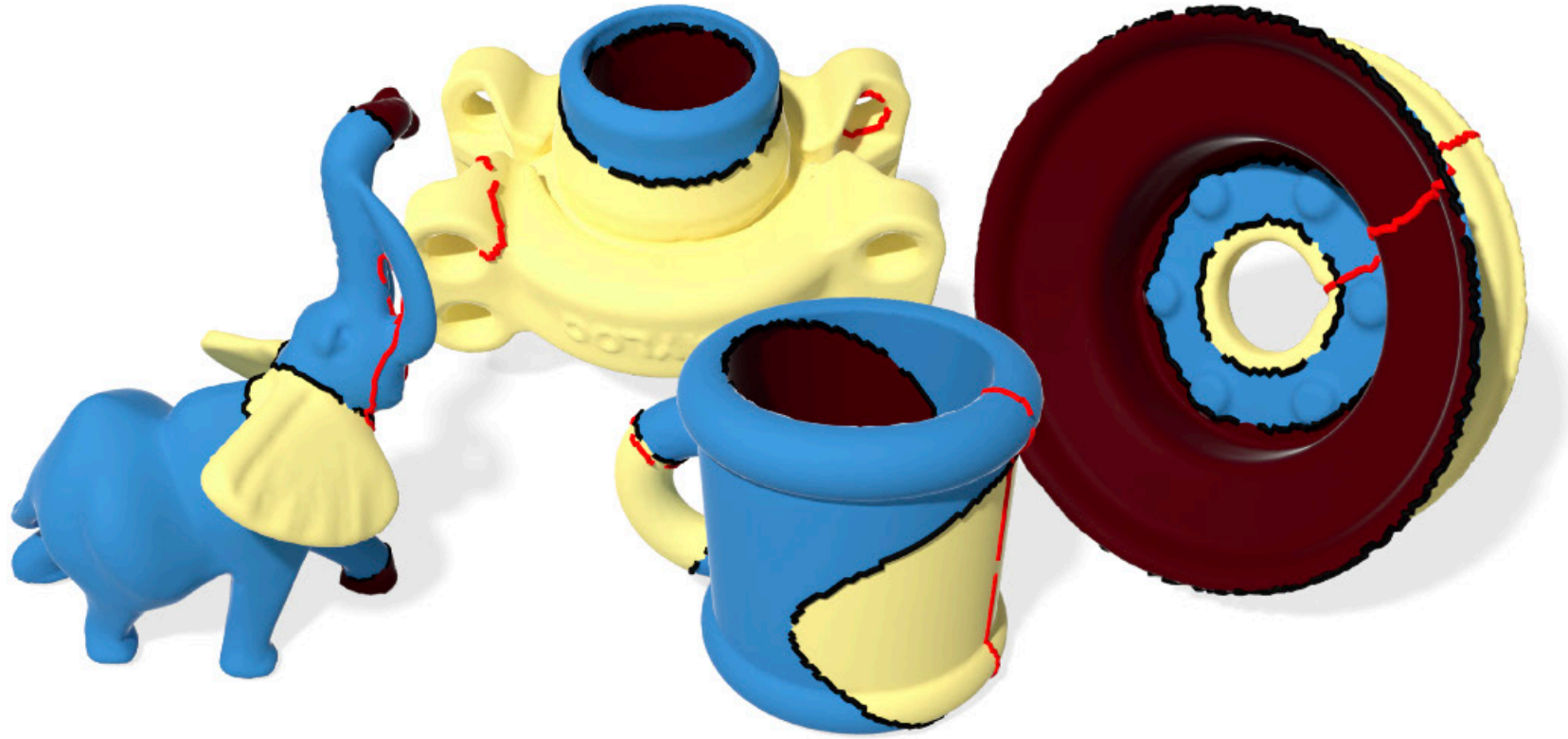


Benchmark setup

(1) Generate random ground-truth regions, take boundaries

[sub-levelsets of low-frequency Laplacian eigenfunctions]

Evaluation



Benchmark setup

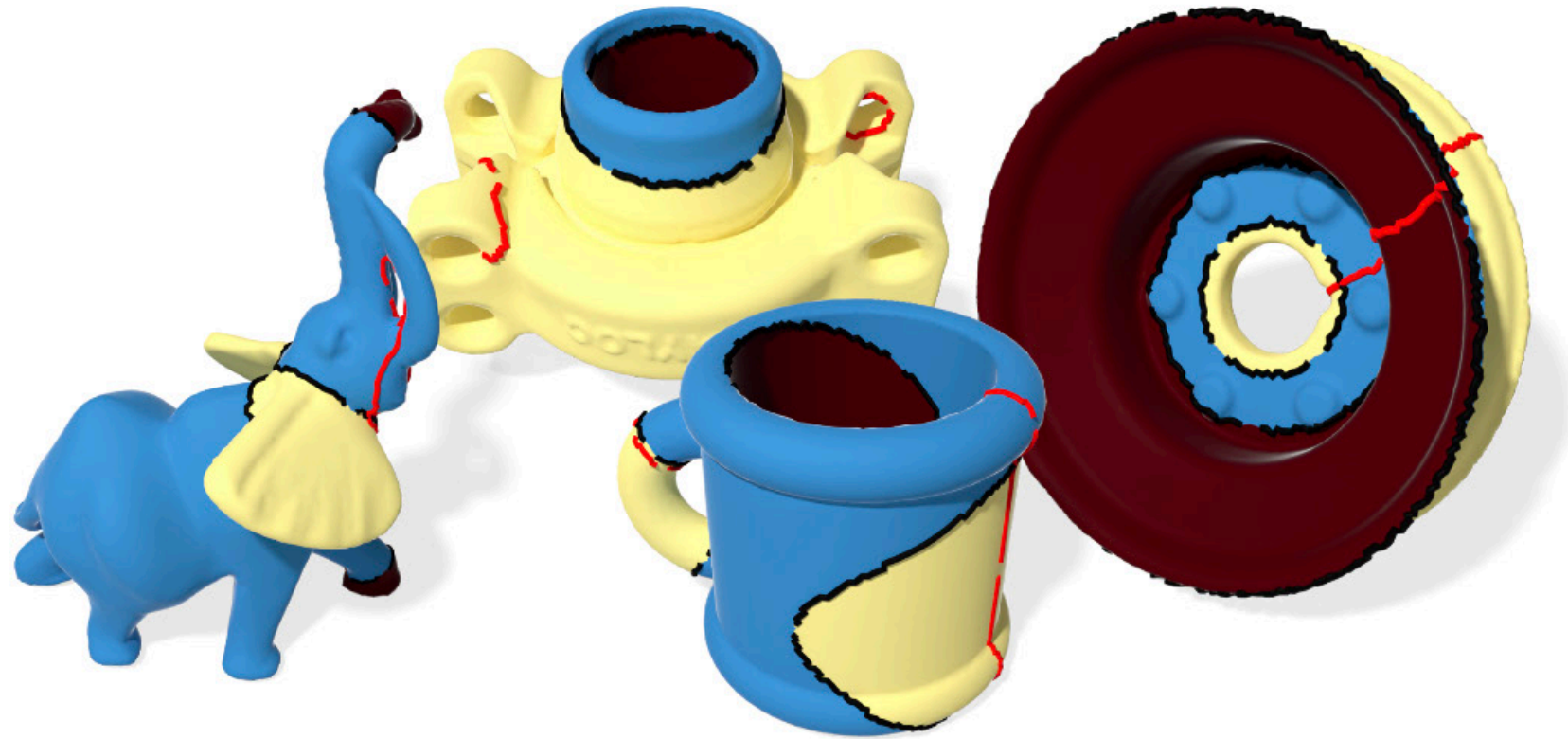
(1) Generate random ground-truth regions, take boundaries

[sub-levelsets of low-frequency Laplacian eigenfunctions]

(2) Add nonbounding loops

[compute homology basis, select random subset]

Evaluation



Benchmark setup

(1) Generate random ground-truth regions, take boundaries

[sub-levelsets of low-frequency Laplacian eigenfunctions]

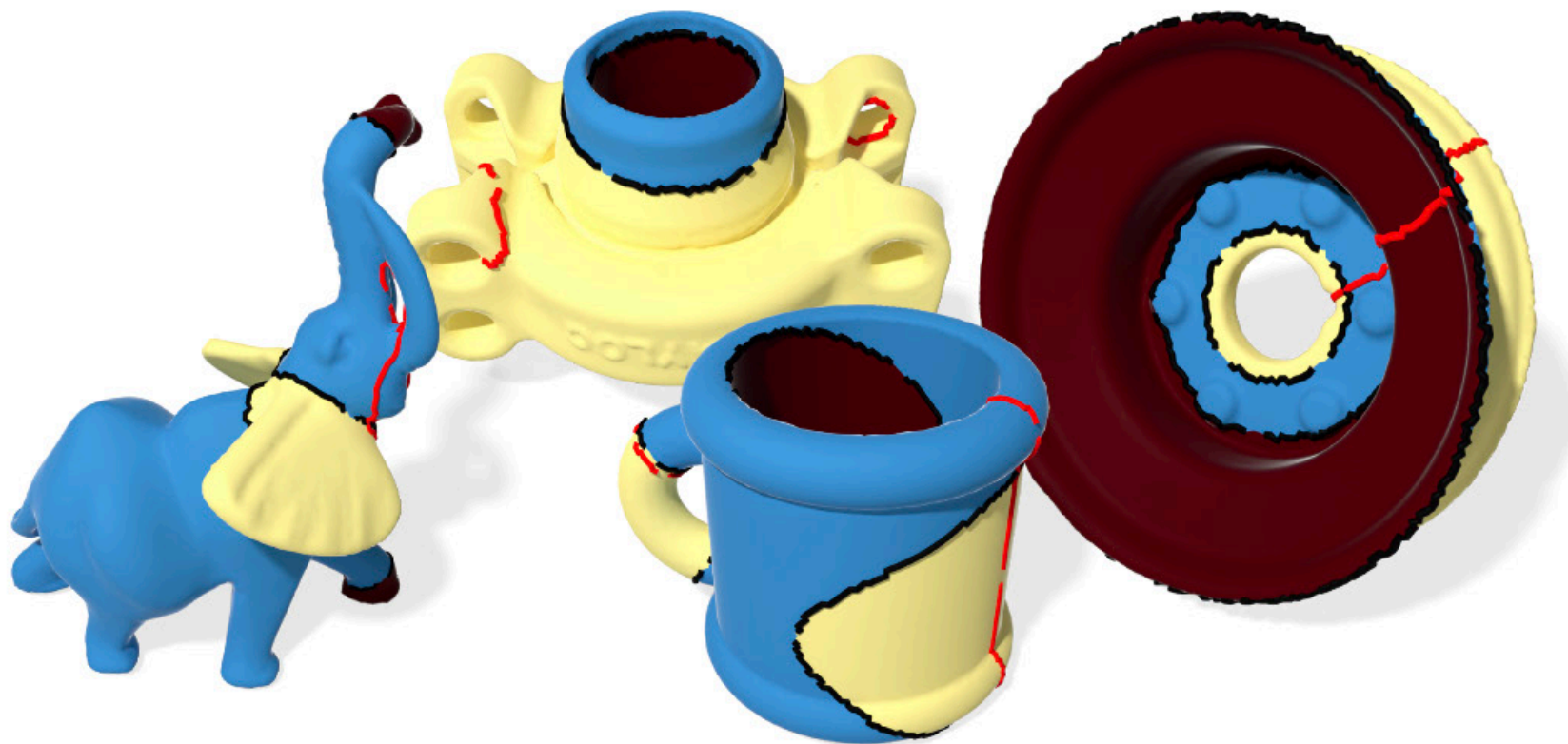
(2) Add nonbounding loops

[compute homology basis, select random subset]

(3) Break up curves

[use random gap & dash size]

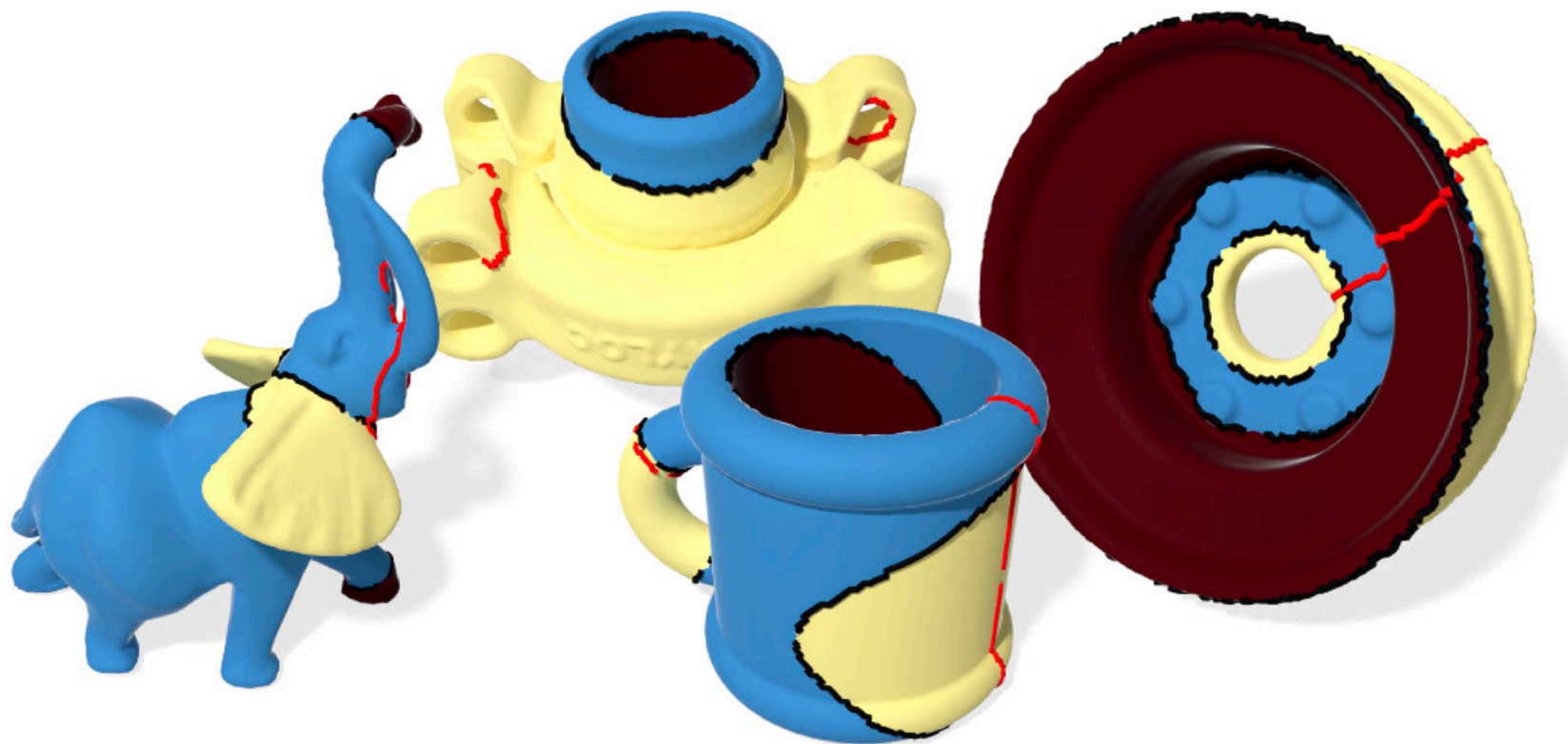
Evaluation



Benchmark setup

- (1) Generate random ground-truth regions, take boundaries *[sub-levelsets of low-frequency Laplacian eigenfunctions]*
- (2) Add nonbounding loops *[compute homology basis, select random subset]*
- (3) Break up curves *[use random gap & dash size]*
- (4) Run SWN; compute % of surface area correctly classified *[shift w to match ground-truth value in an arbitrary face]*

Evaluation



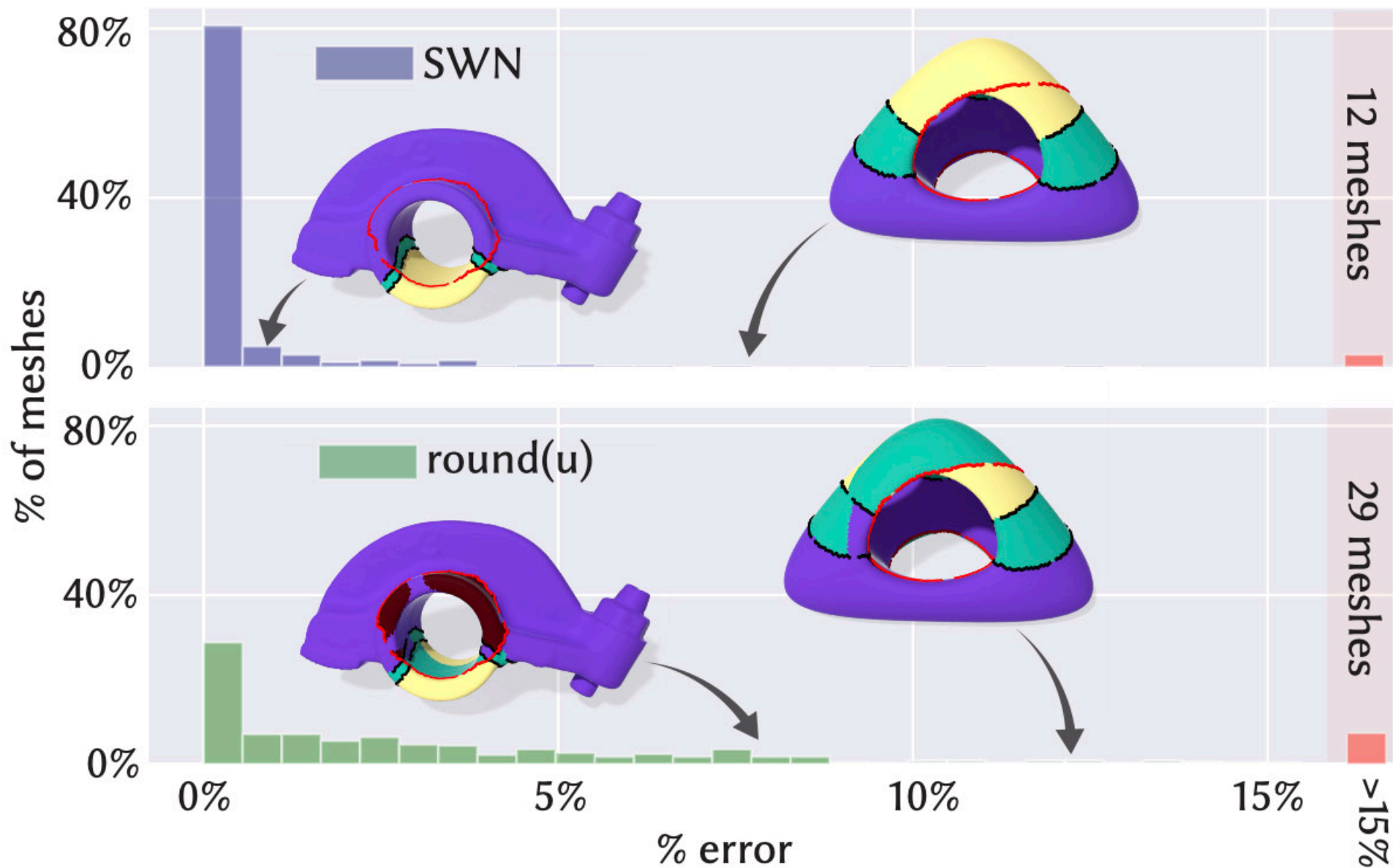
Benchmark setup

- (1) Generate random ground-truth regions, take boundaries *[sub-levelsets of low-frequency Laplacian eigenfunctions]*
- (2) Add nonbounding loops *[compute homology basis, select random subset]*
- (3) Break up curves *[use random gap & dash size]*
- (4) Run SWN; compute % of surface area correctly classified *[shift w to match ground-truth value in an arbitrary face]*

934 total test cases, 451 multiply-connected

Evaluation

Success rate on nontrivial surfaces



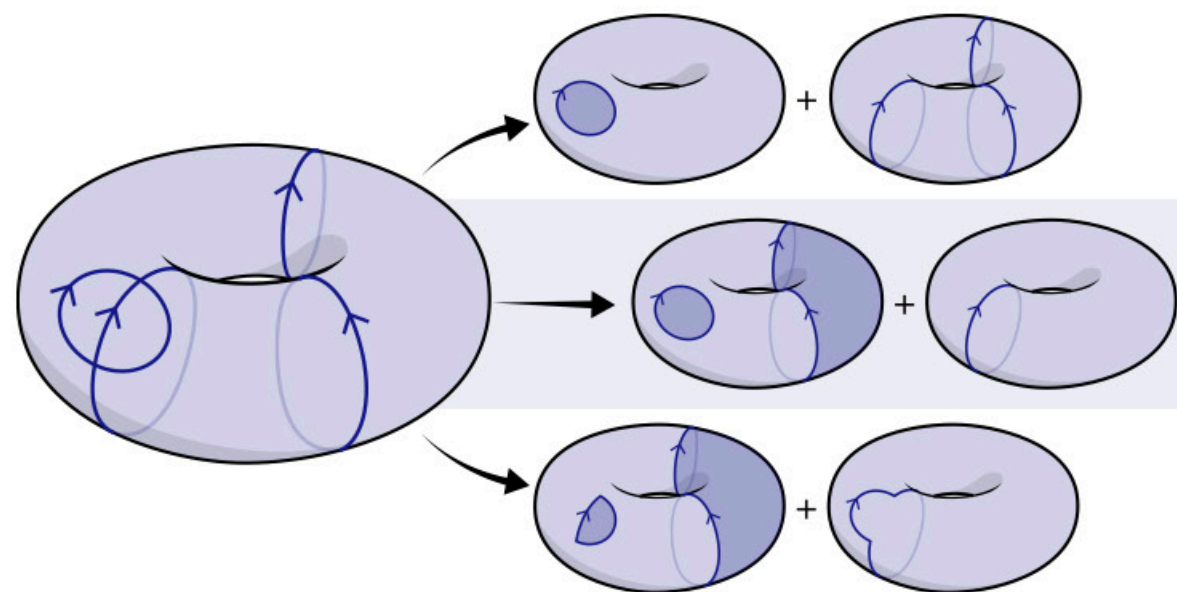
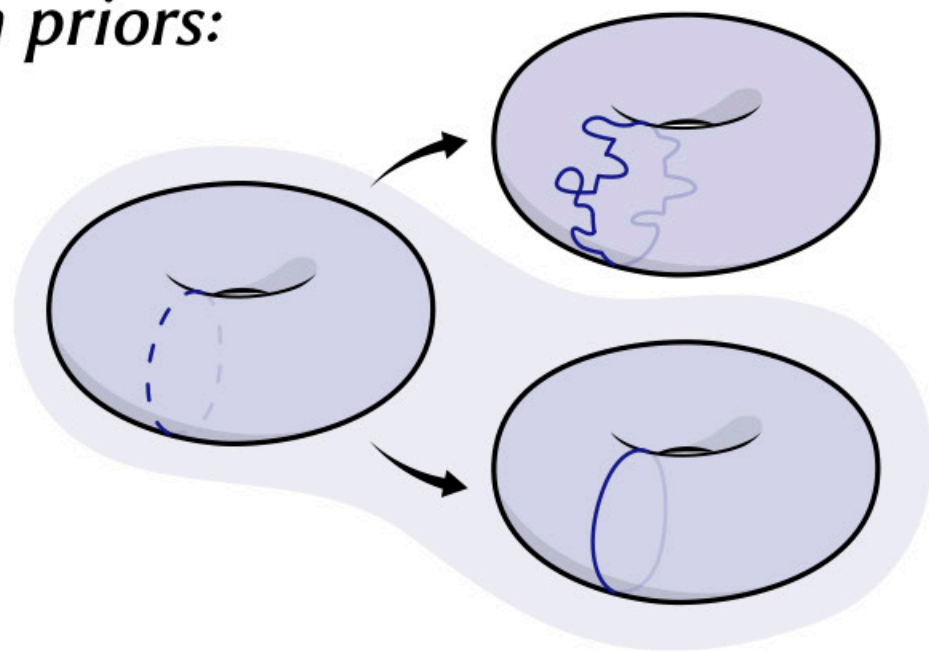
Ours

PSR/GWN

Much less accurate
without homology
processing

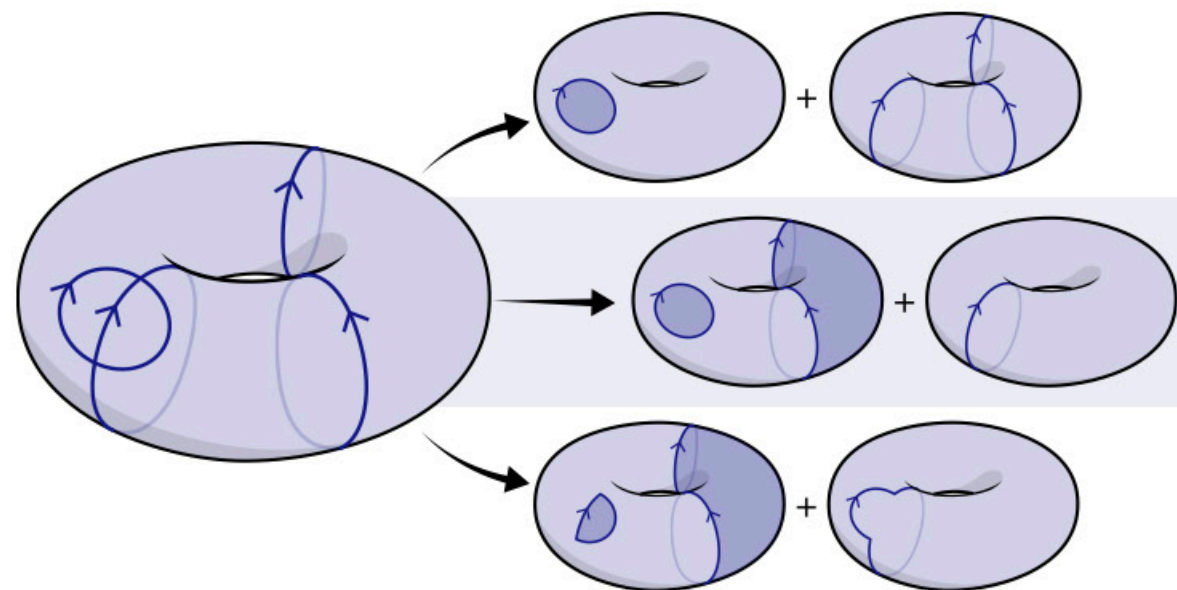
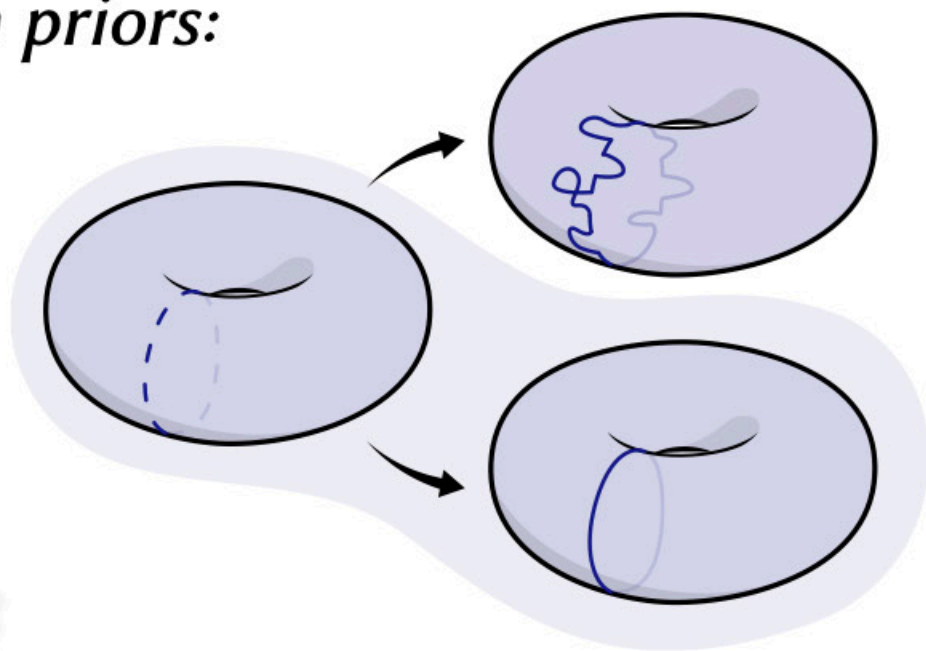
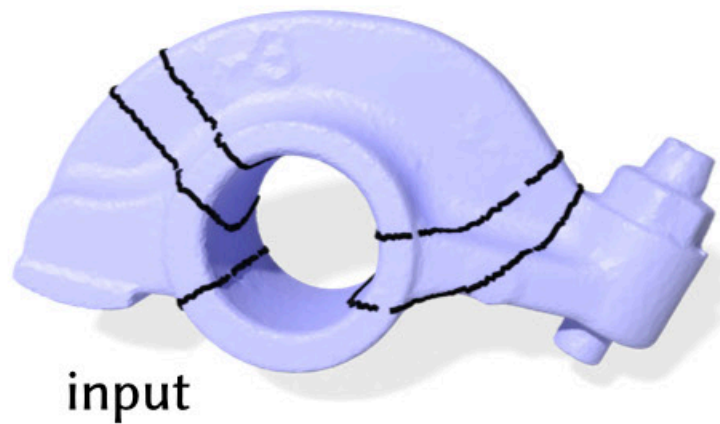
“Failure” cases

Recall shortest-length priors:



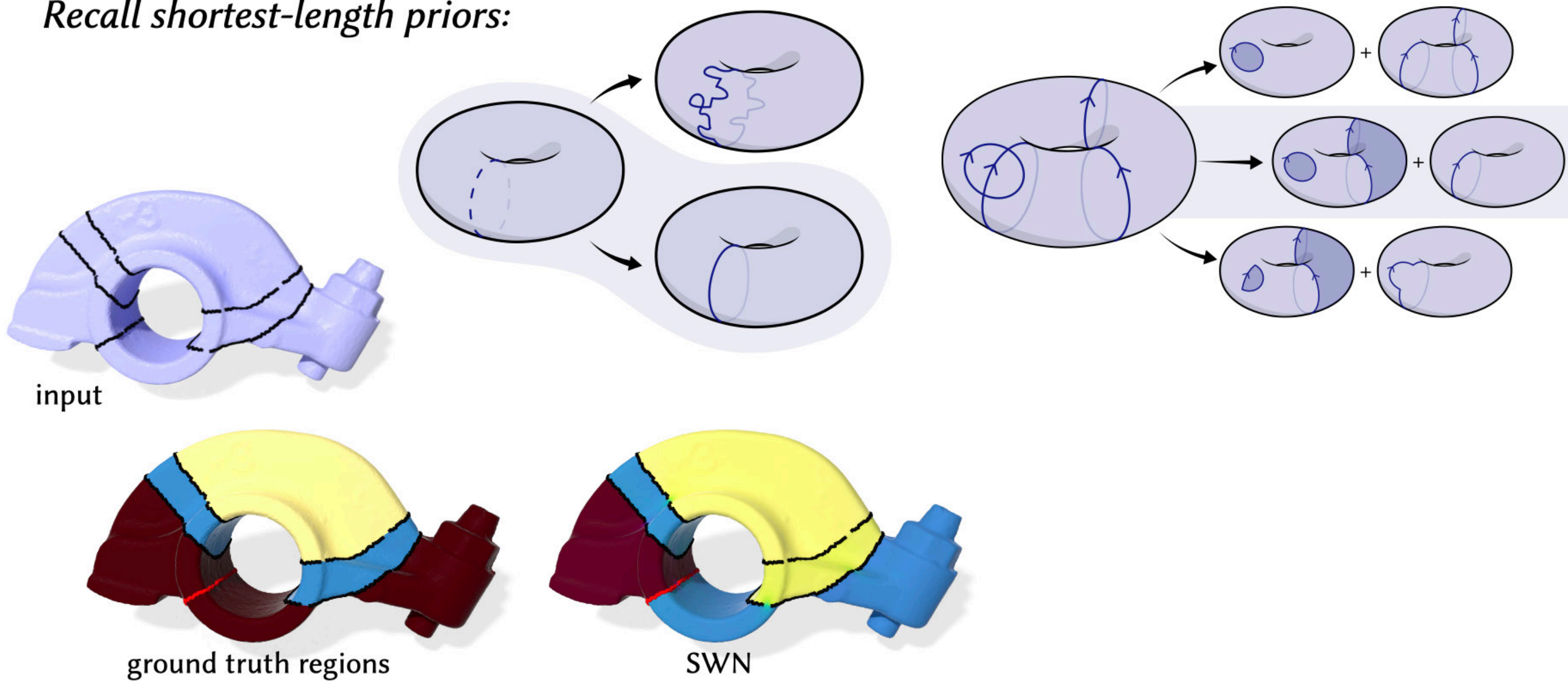
“Failure” cases

Recall shortest-length priors:



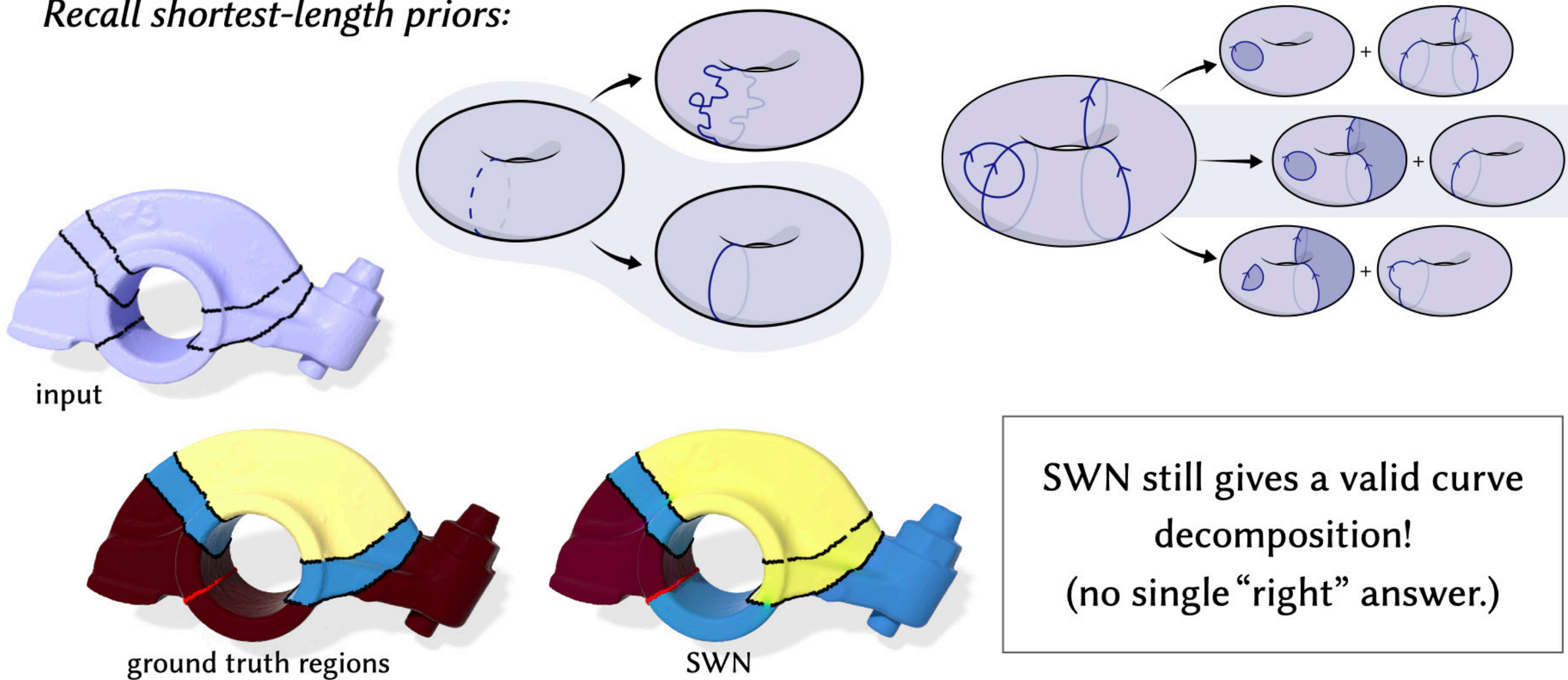
“Failure” cases

Recall shortest-length priors:

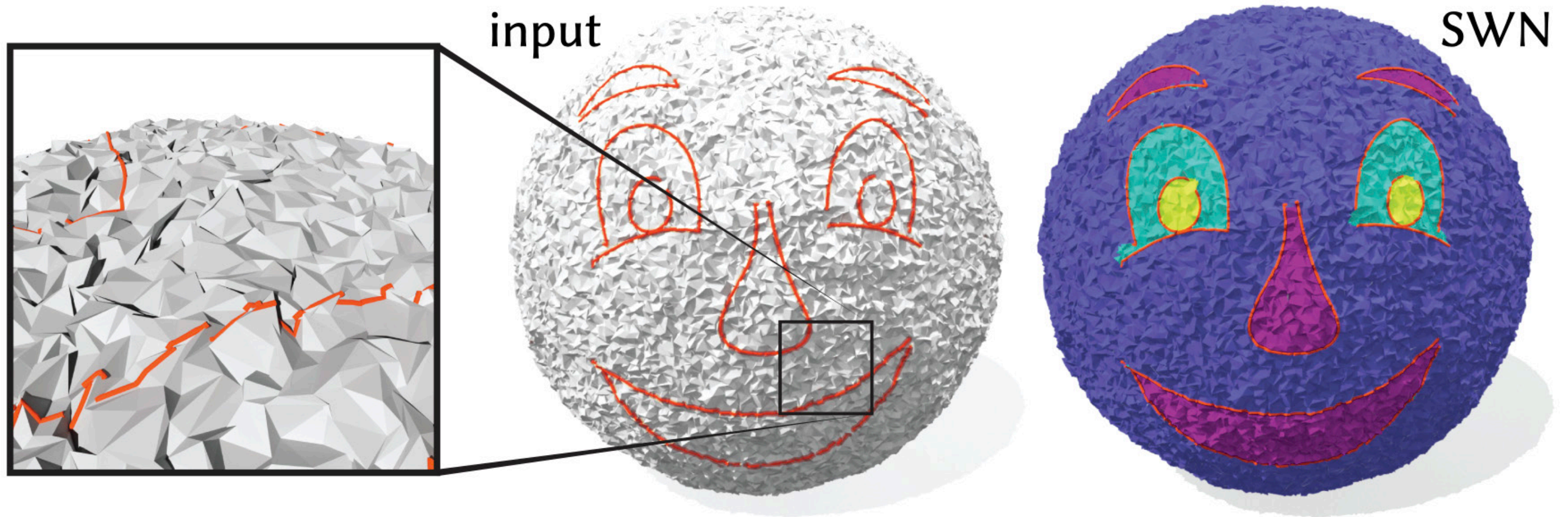


“Failure” cases

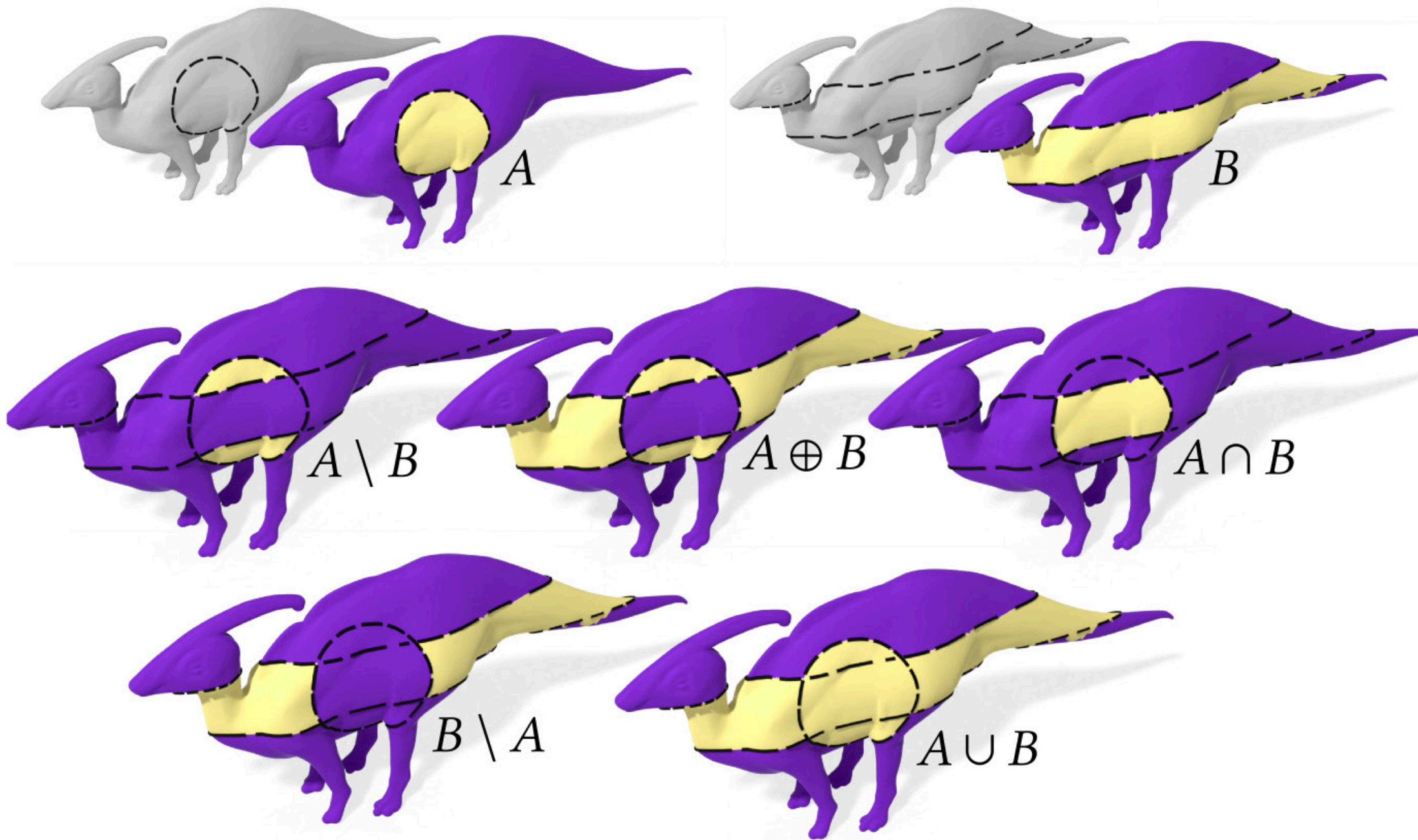
Recall shortest-length priors:



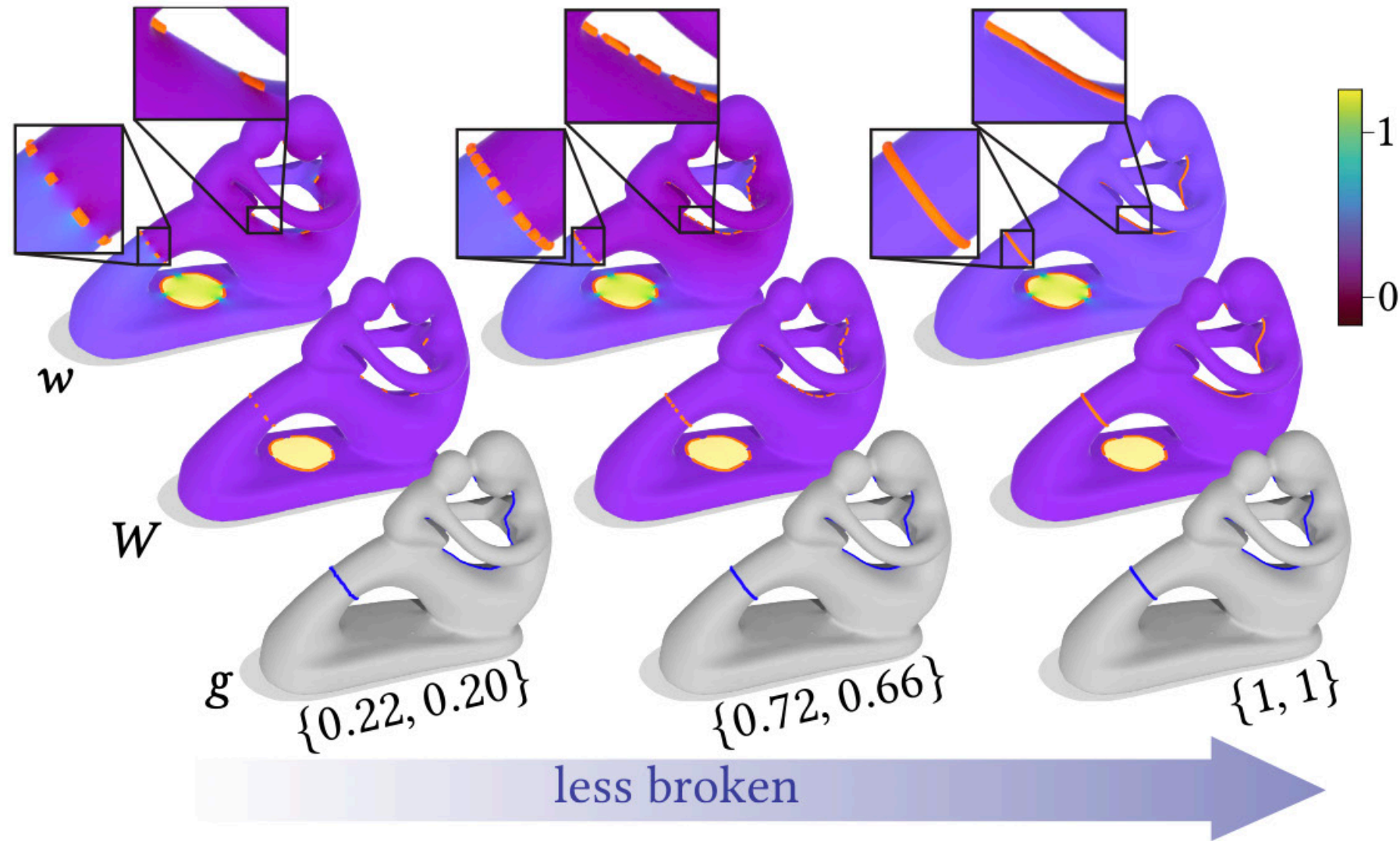
Surface sketching



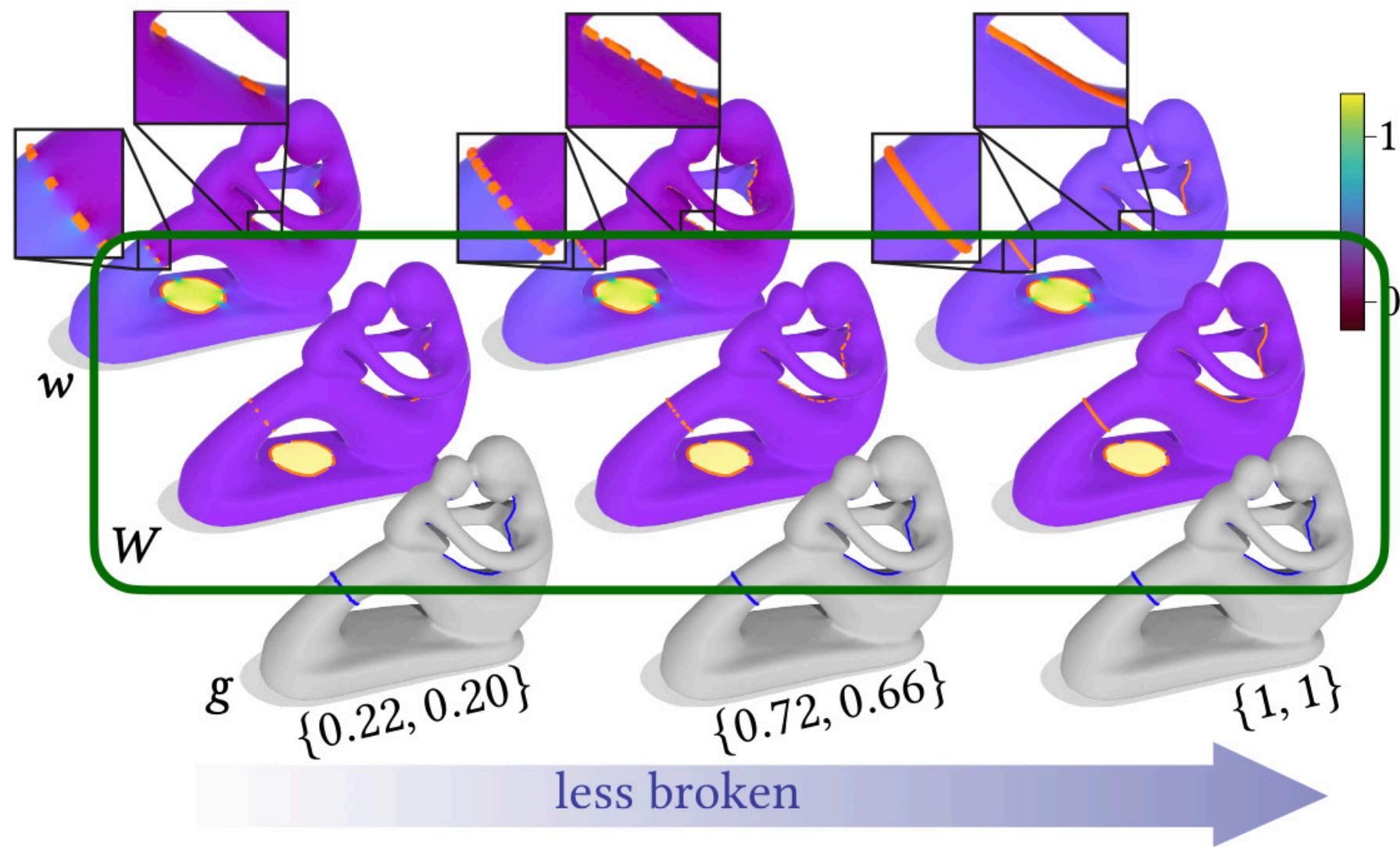
Booleans



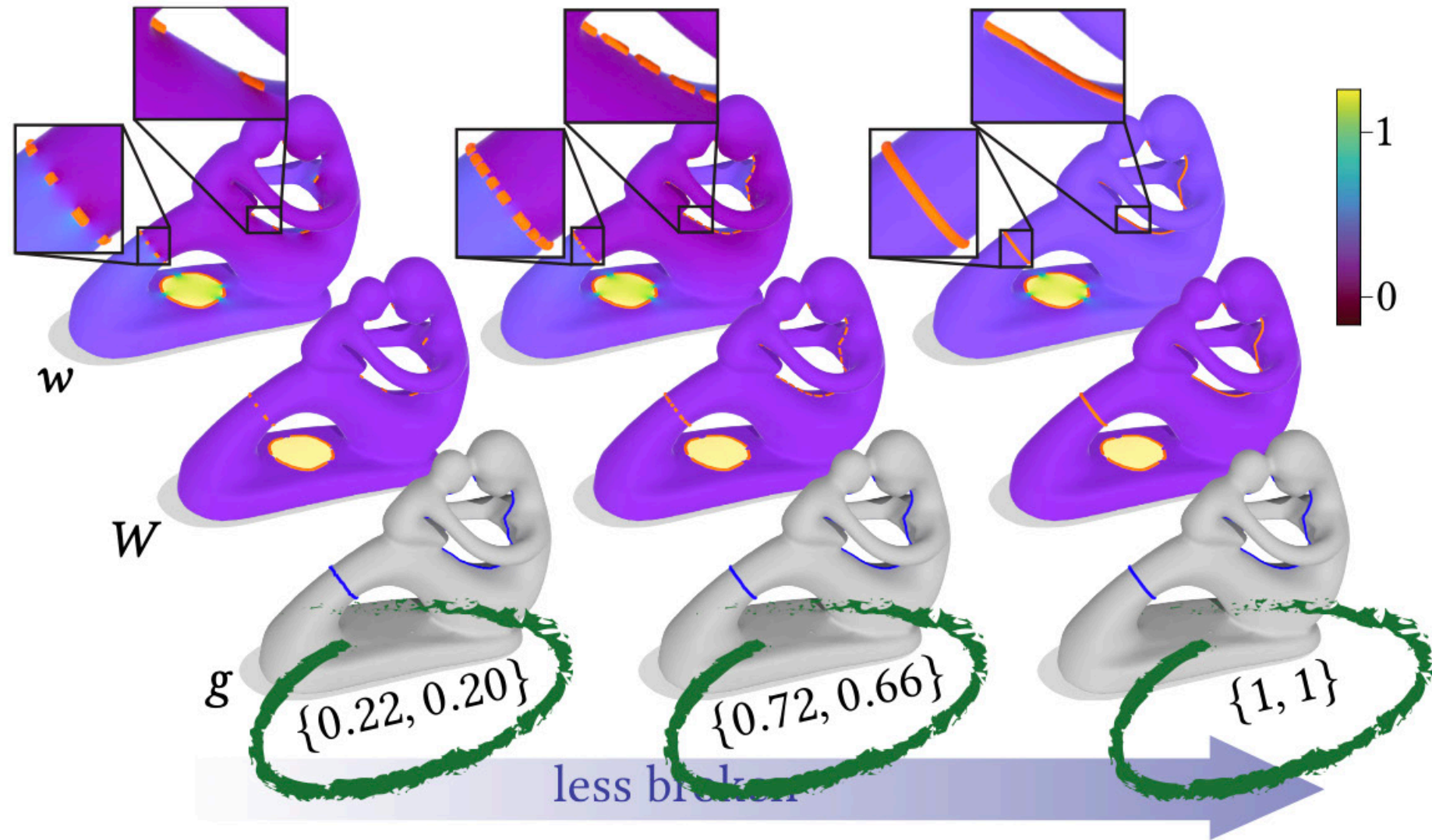
Curve decomposition



Curve decomposition

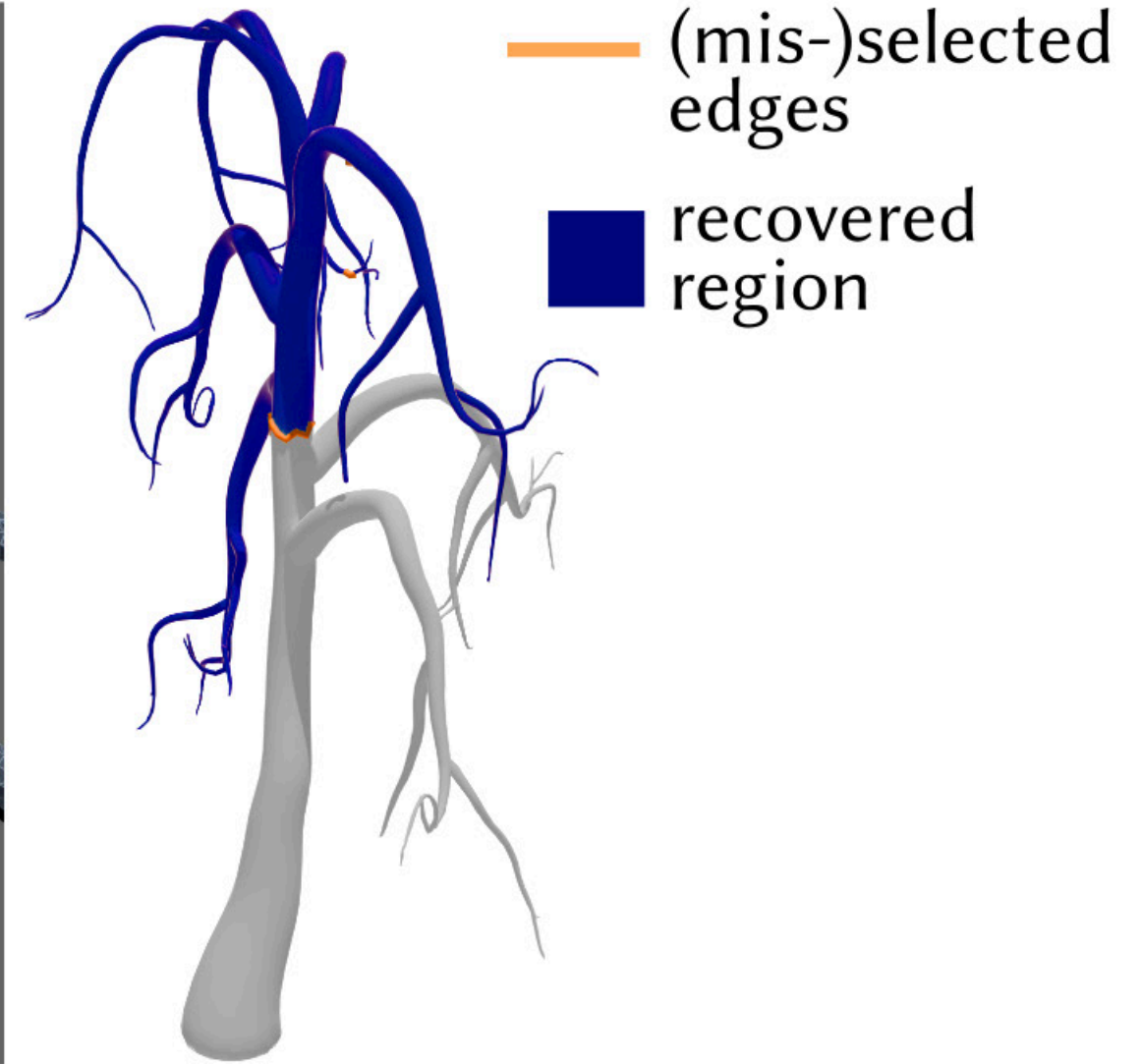
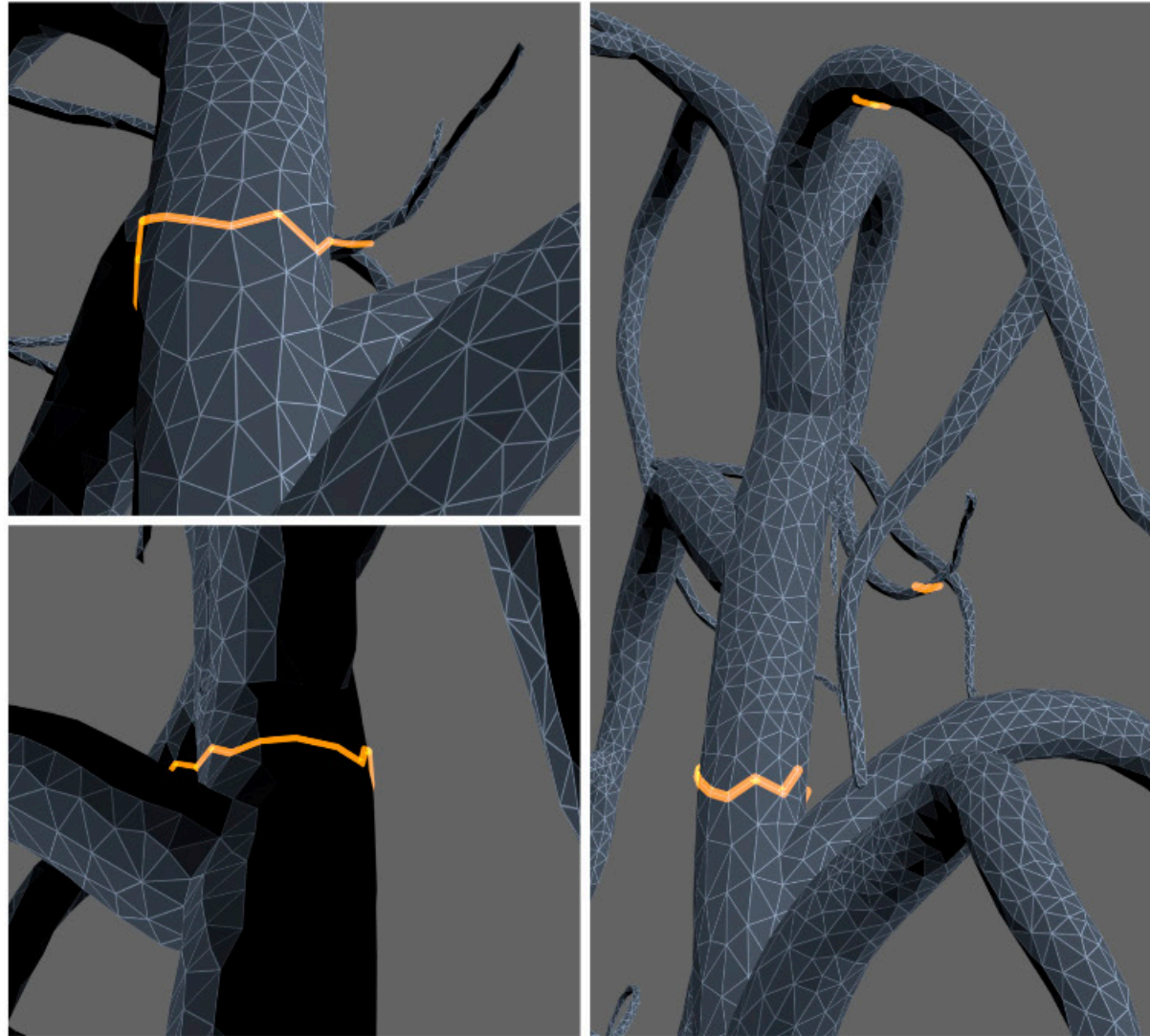


Curve decomposition



g values give SWN's confidence in nonbounding loops

Region selection — robustness

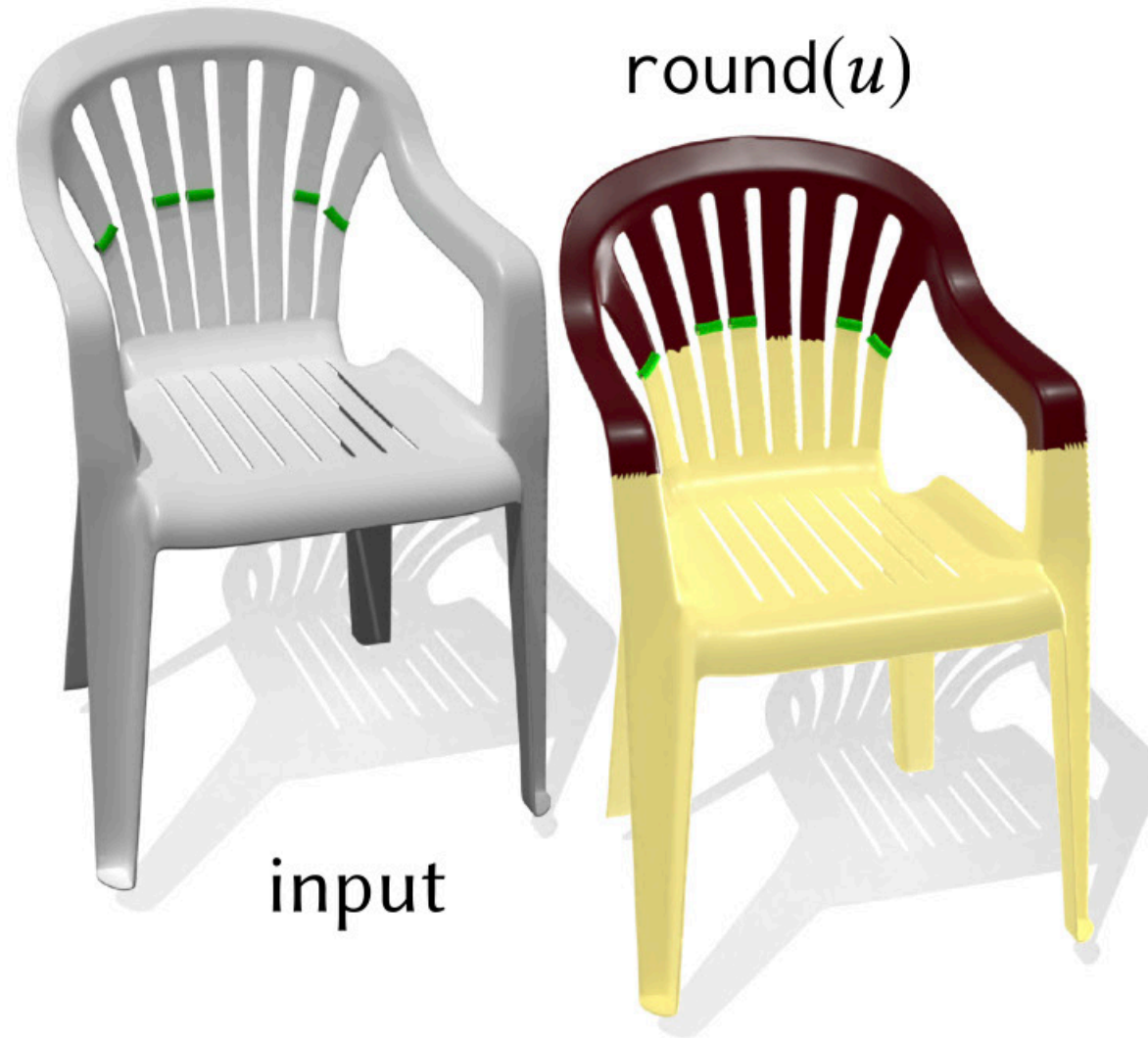


Region selection — automatic completion



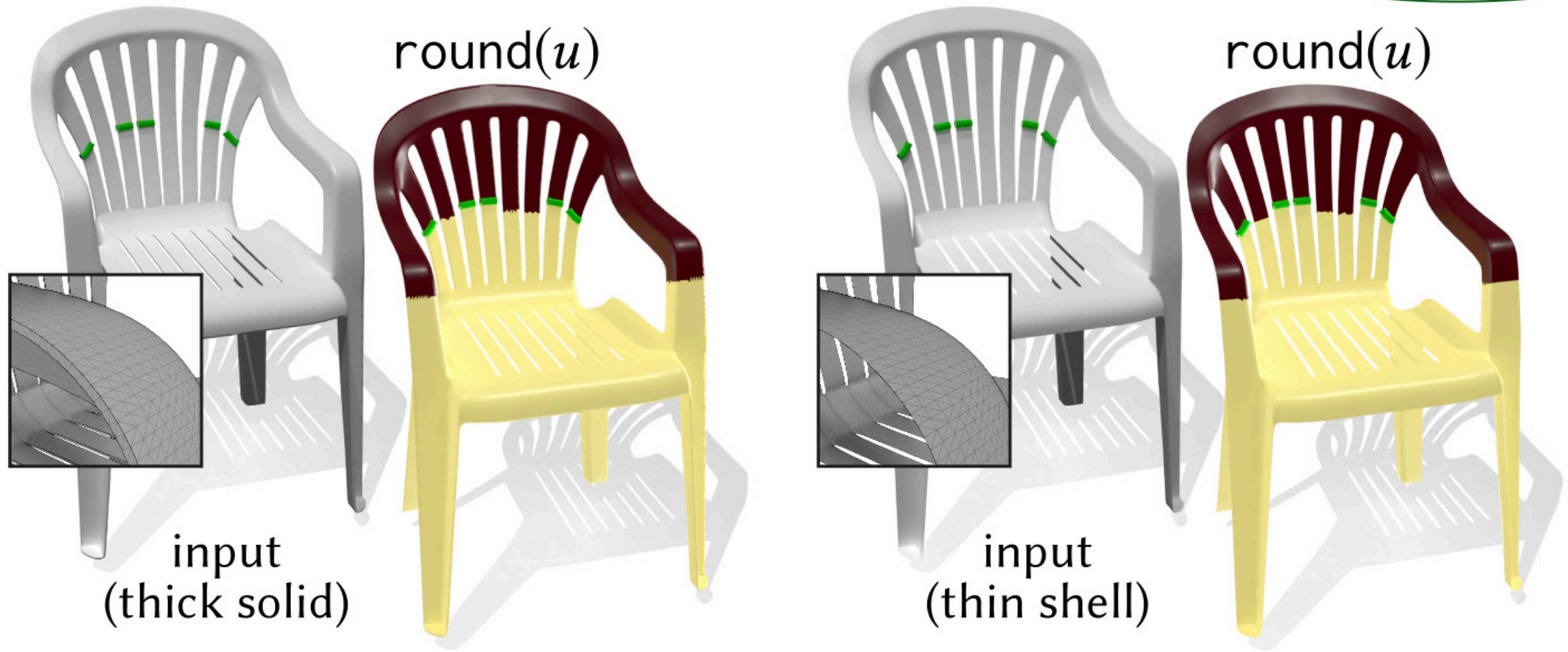
Region selection – automatic completion

using “ghost loops”
to *aid* segmentation



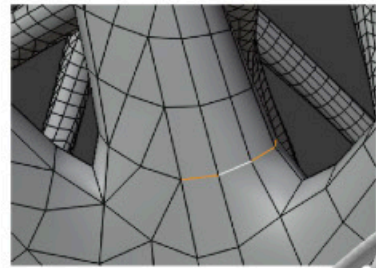
Region selection – automatic completion

using “ghost loops”
to *aid* segmentation

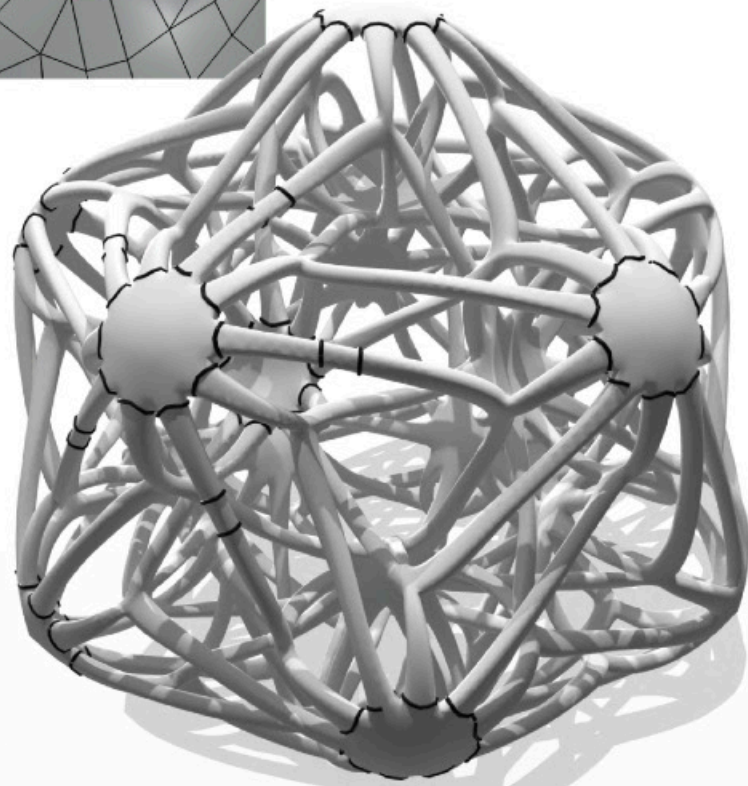


Region selection – automatic completion

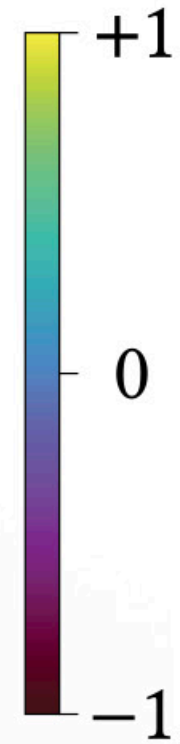
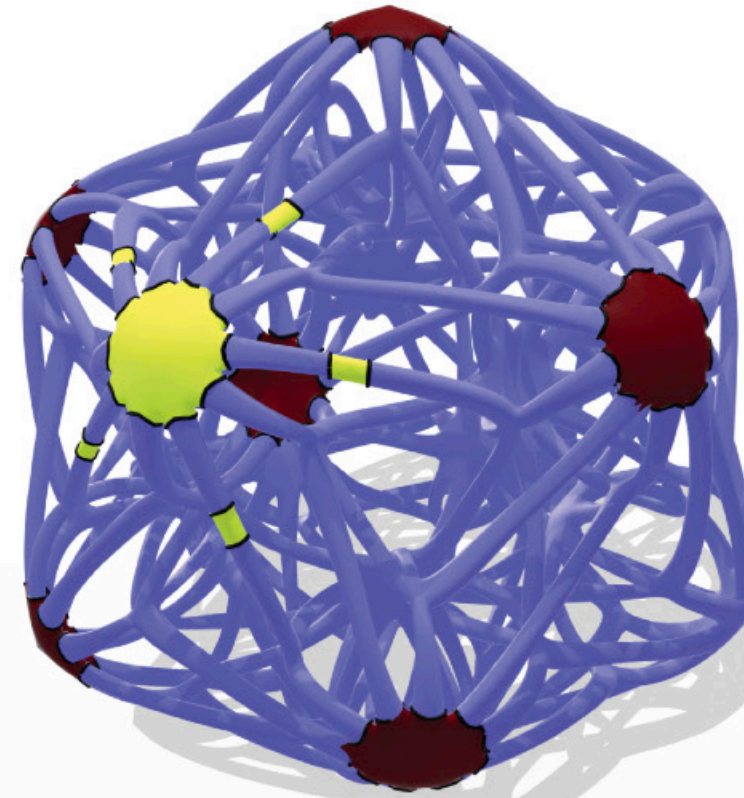
using “ghost loops”
to *aid* segmentation



incomplete edge
loop selection

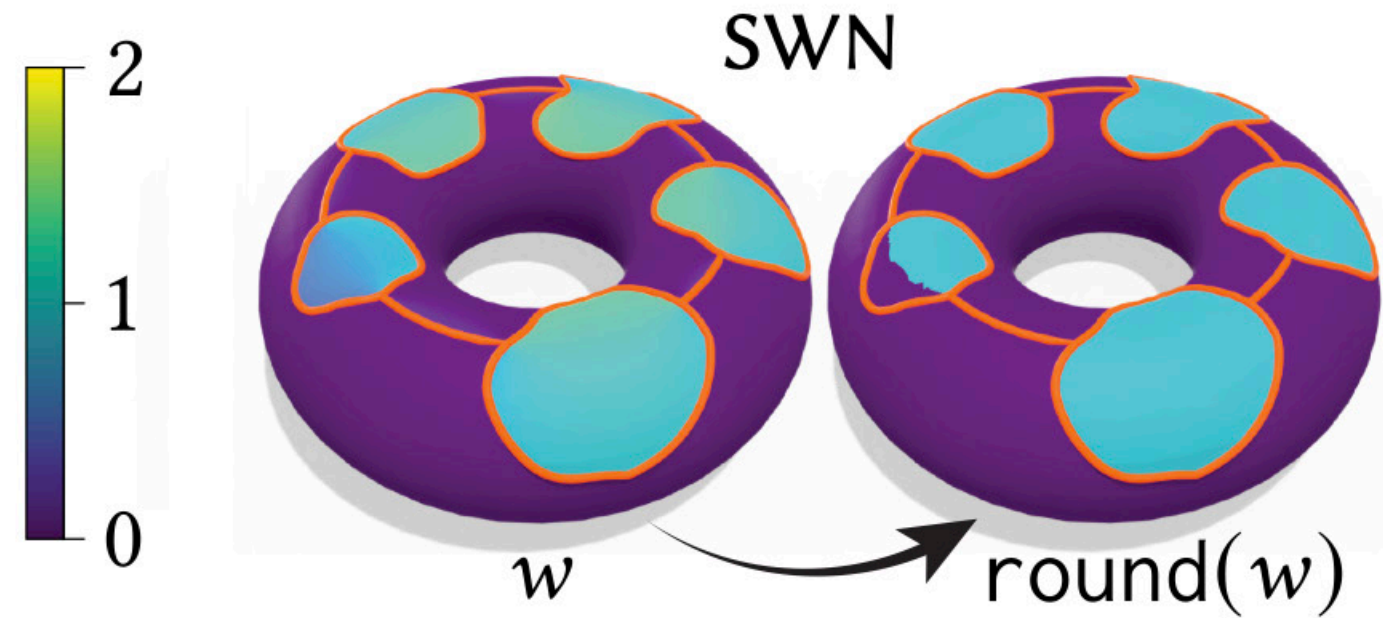


$\text{round}(u)$

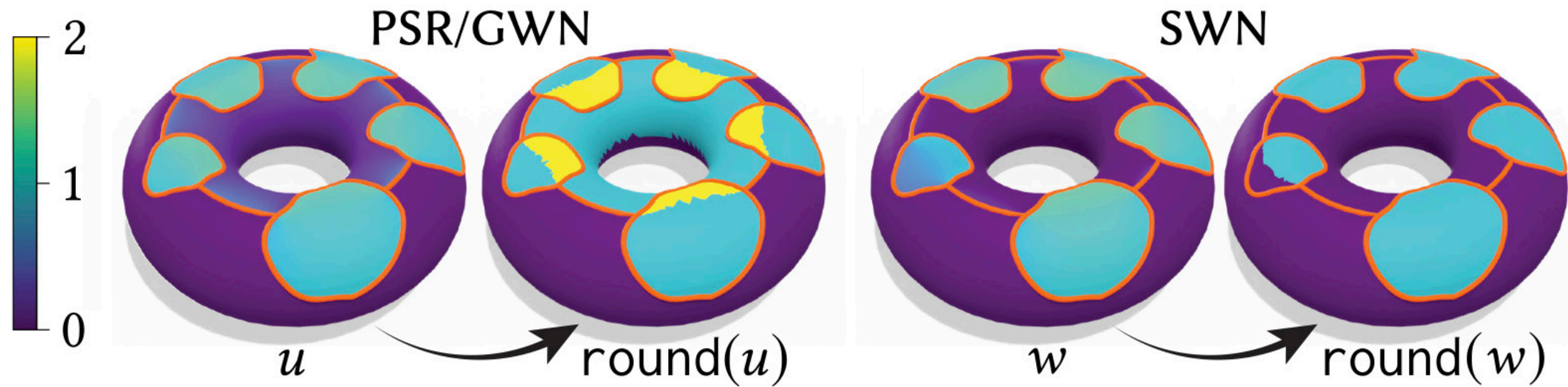


LIMITATIONS & FUTURE WORK

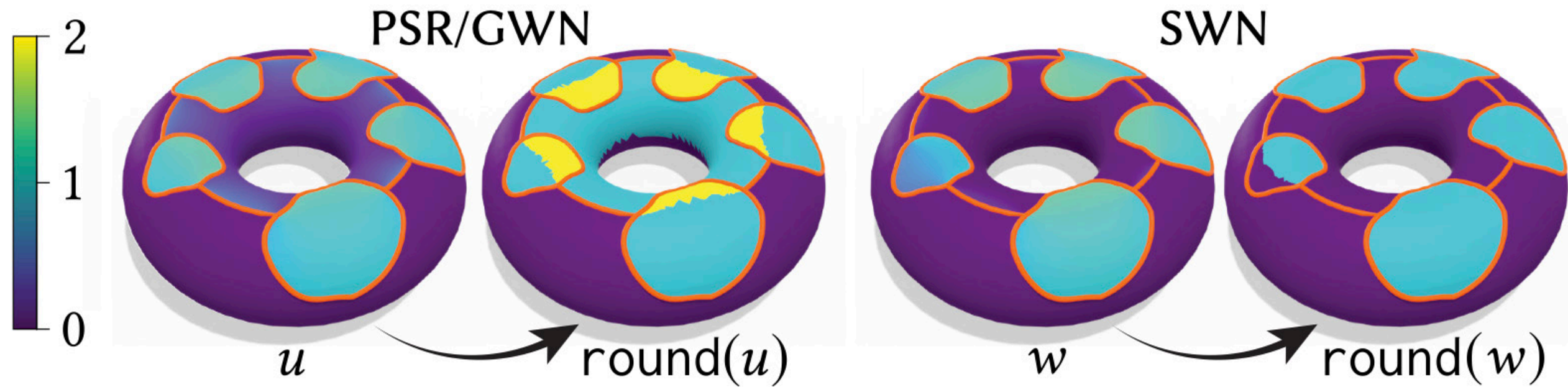
Contouring is sometimes counterintuitive



Contouring is sometimes counterintuitive



Contouring is sometimes counterintuitive



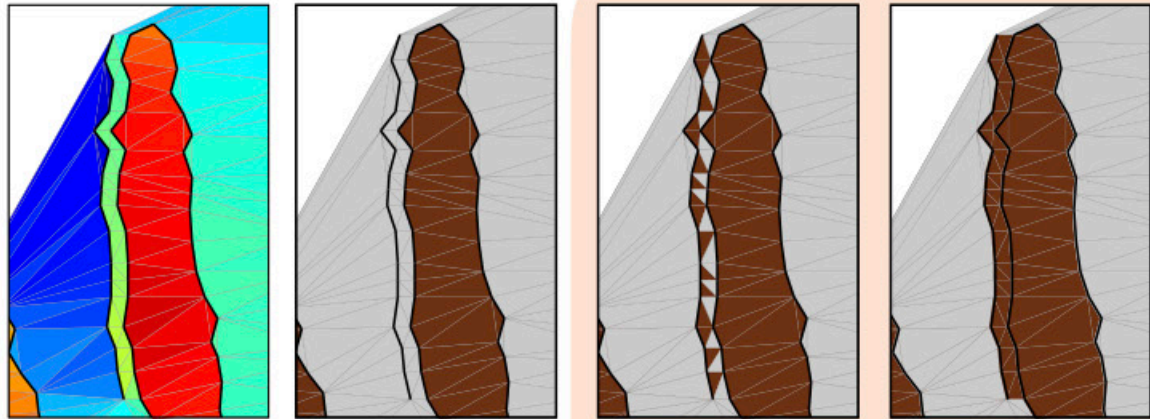
Euclidean winding number also struggles

Jacobson et al. suggest a graphcut algorithm for contouring:

Robust Inside-Outside Segmentation using Generalized Winding Numbers
Jacobson, Kavan, Sorkine-Hornung (2013)

Euclidean winding number also struggles

Jacobson et al. suggest a graphcut algorithm for contouring:



GWN

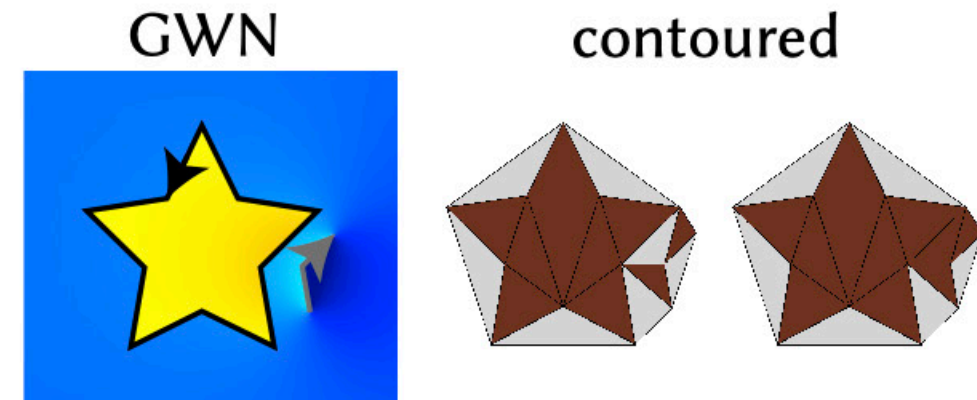
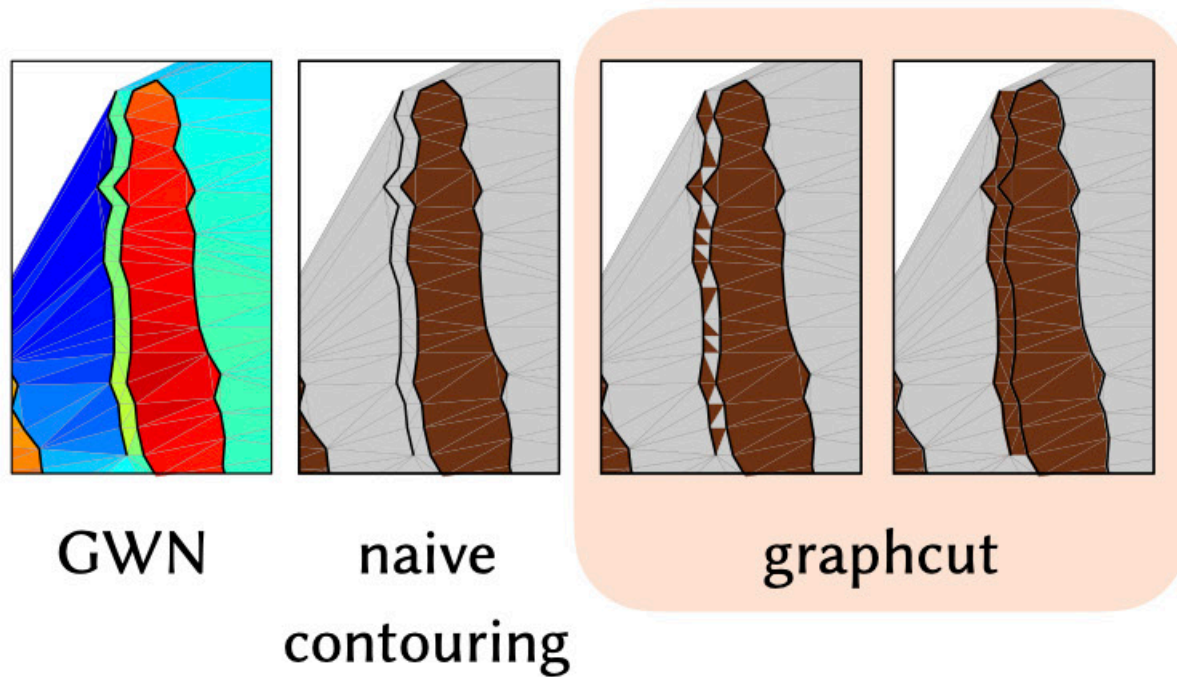
naive
contouring

graphcut

Robust Inside-Outside Segmentation using Generalized Winding Numbers
Jacobson, Kavan, Sorkine-Hornung (2013)

Euclidean winding number also struggles

Jacobson et al. suggest a graphcut algorithm for contouring:



Still not perfect!

Robust Inside-Outside Segmentation using Generalized Winding Numbers
Jacobson, Kavan, Sorkine-Hornung (2013)

Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$

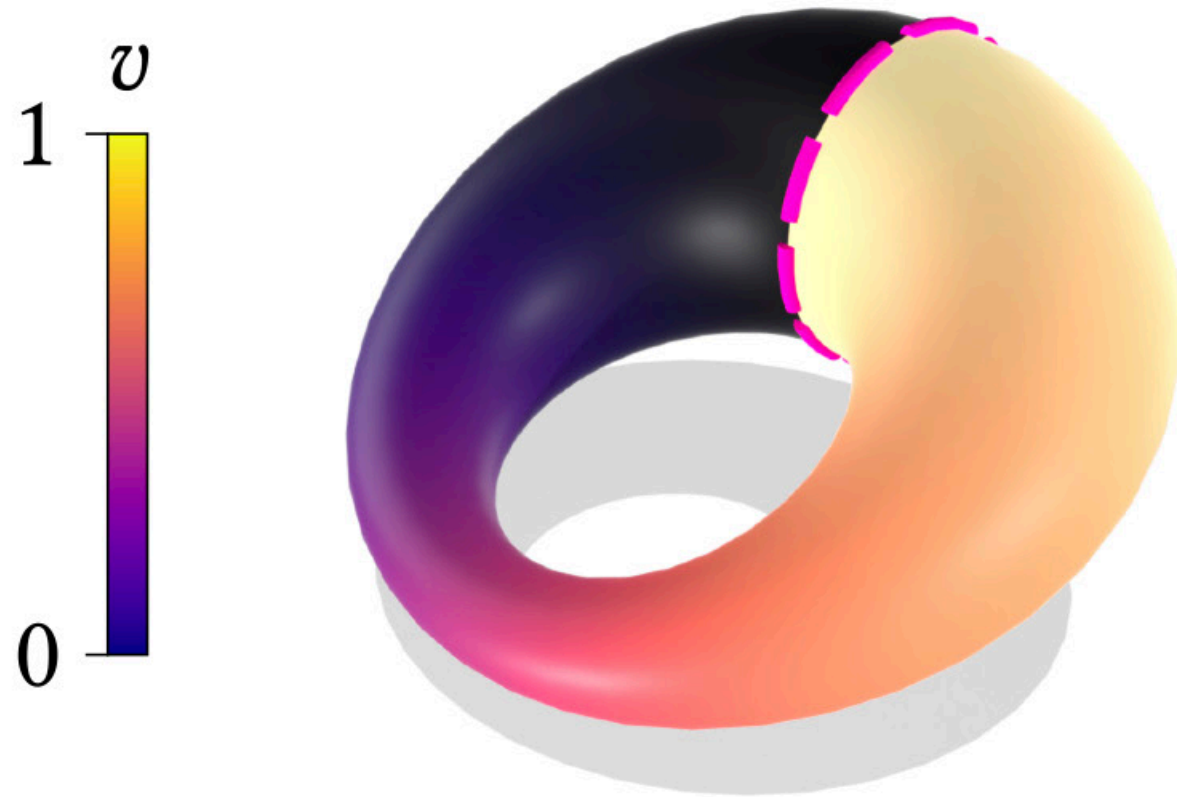
Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$ *encourage jumps across Γ*

Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$

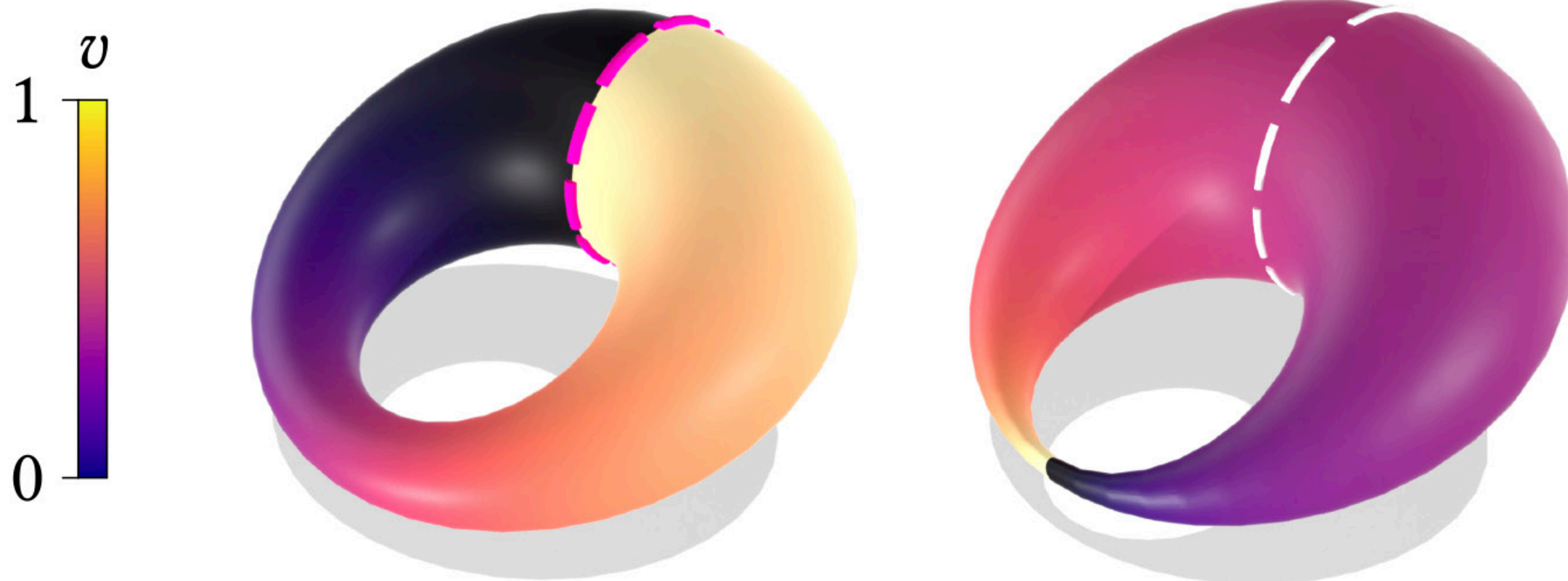
*encourage jumps
across Γ*



Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$

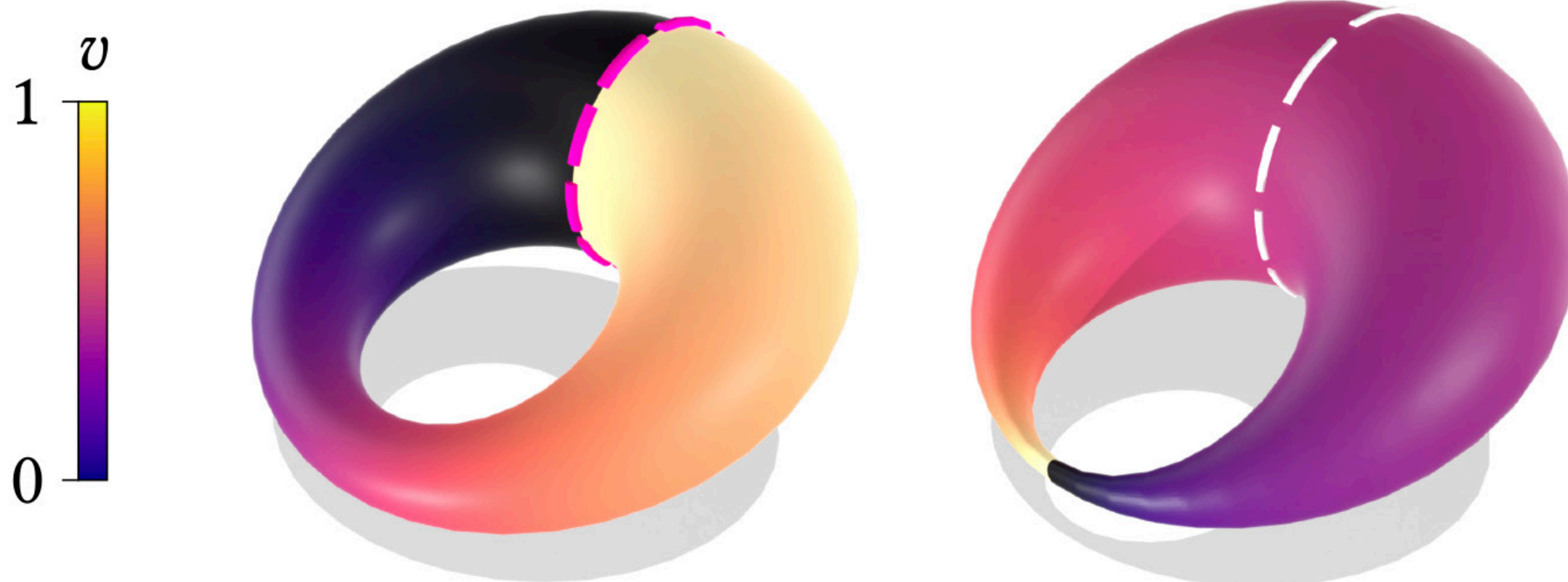
*encourage jumps
across Γ*



Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$

*encourage jumps
across Γ*



Can always adversarially increase a handle's taper.

Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$ *encourage jumps across Γ*



Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$

encourage jumps
across Γ

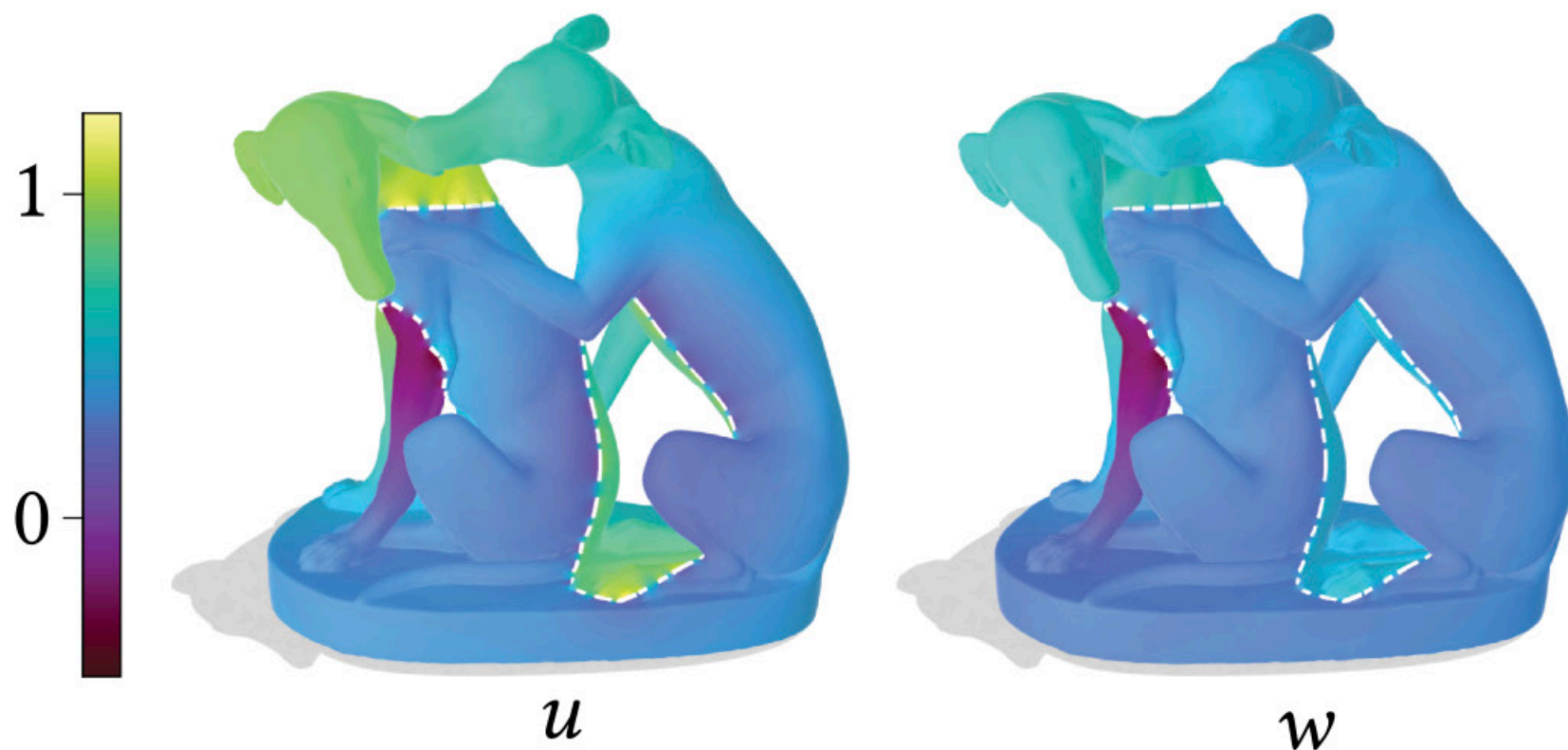


But given a *fixed* mesh, we will recover the correct solution as gaps $\rightarrow 0$
(and appropriate choice of ε).

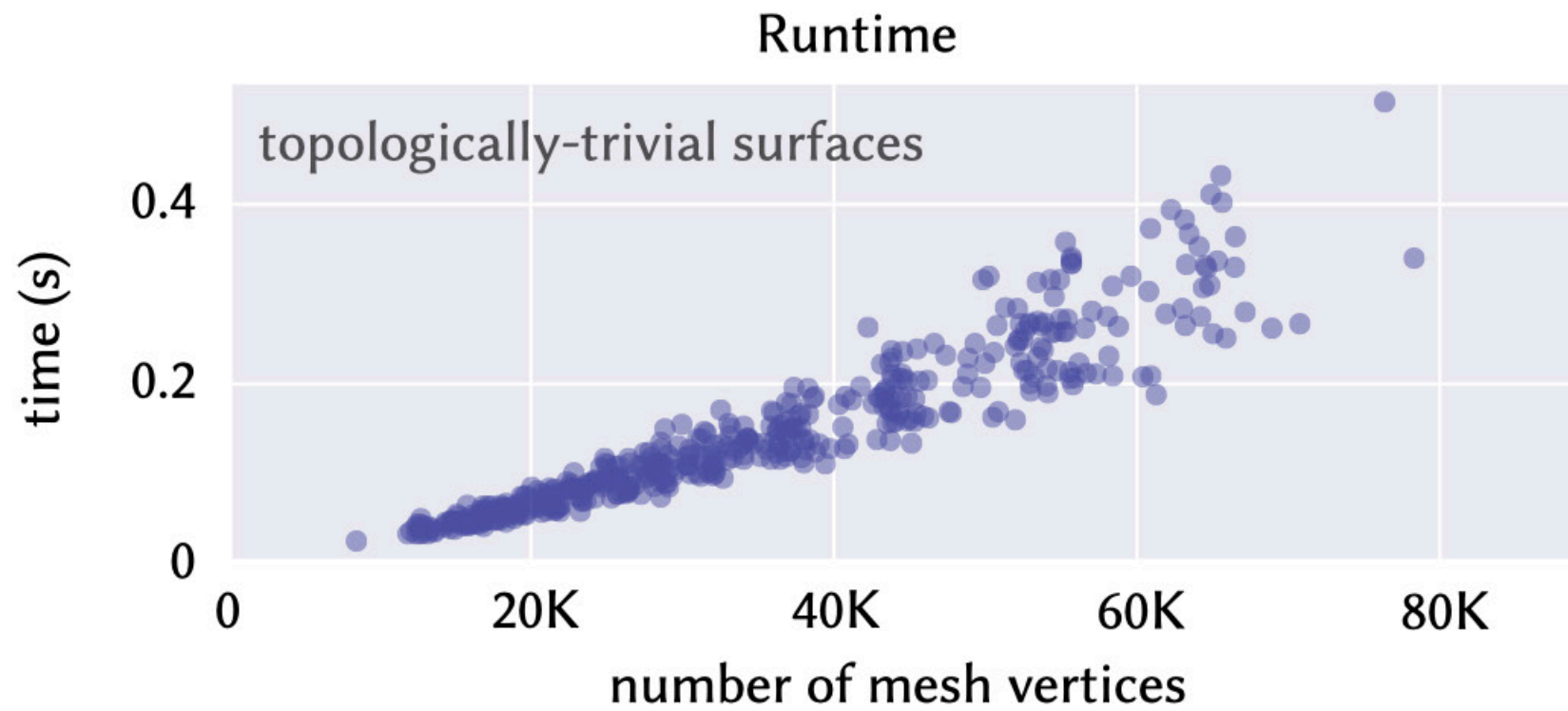
Unexpected nonbounding loops

Recall objective function: $\min_{v: M \rightarrow \mathbb{R}} \int |\text{the jumps not across } \Gamma| + \varepsilon \int |\text{the jumps across } \Gamma|$

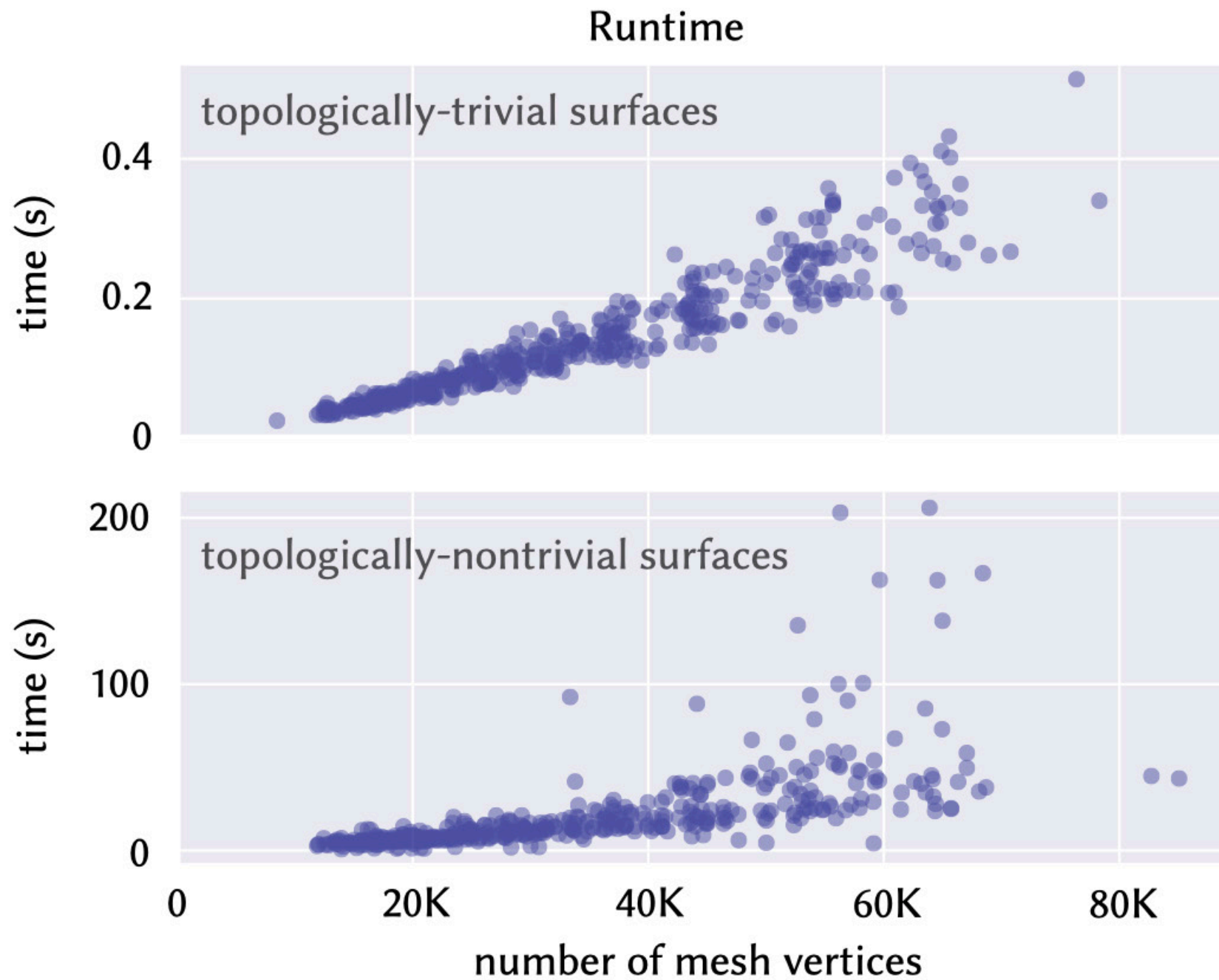
*encourage jumps
across Γ*



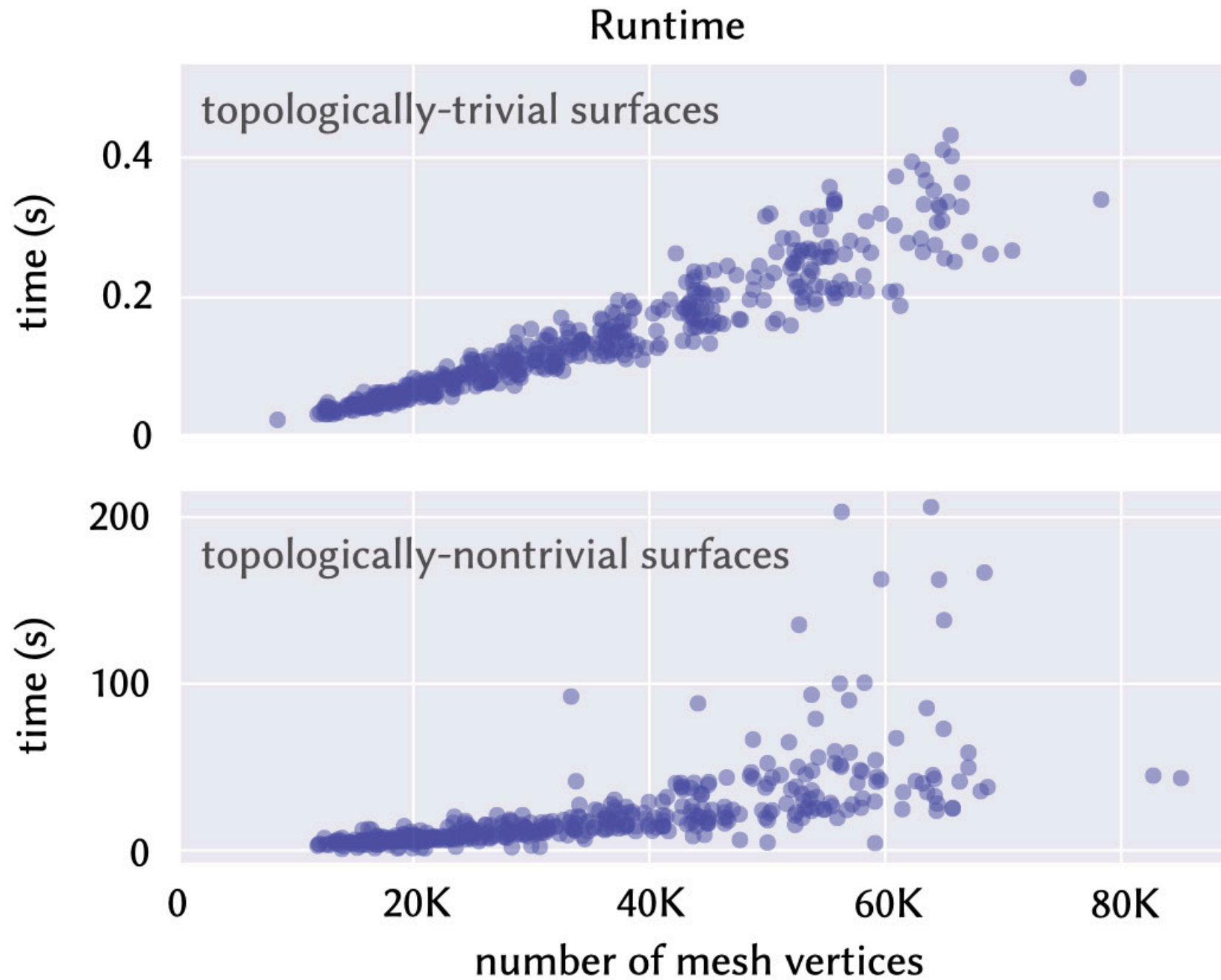
Performance



Performance



Performance



Computation dominated by linear program.

Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

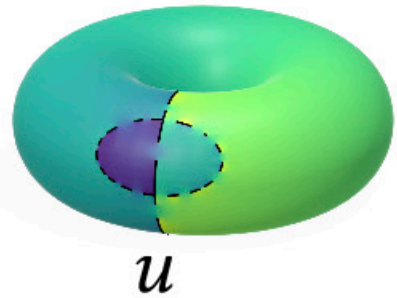
1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .

Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .

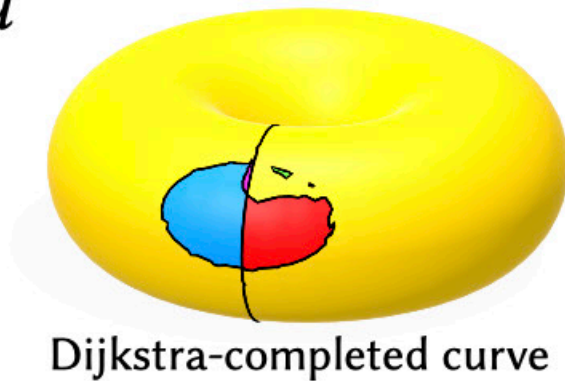
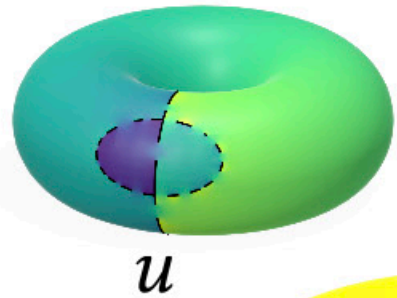


Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .

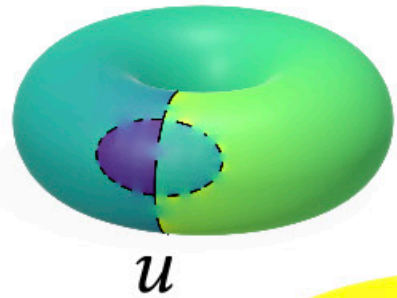


Reduced-size linear program

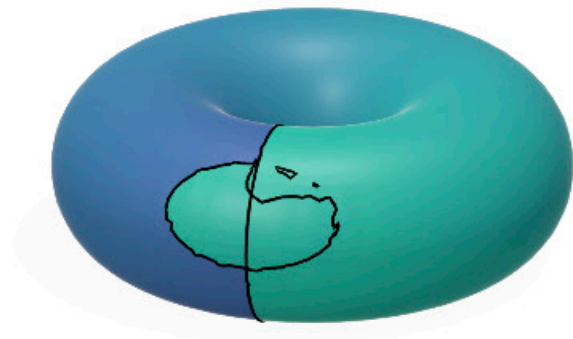
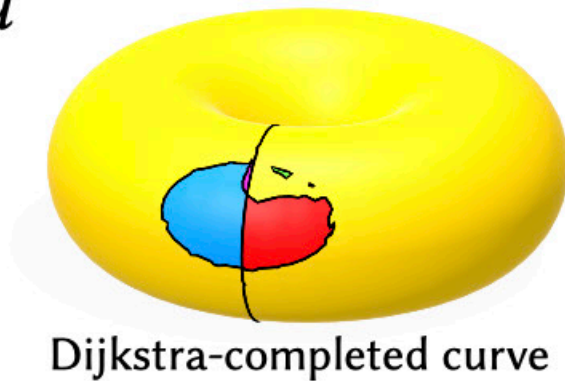
Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .



Integrate γ locally within each connected component



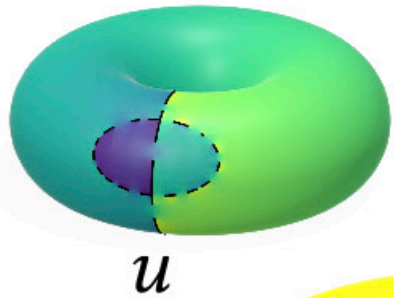
Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .

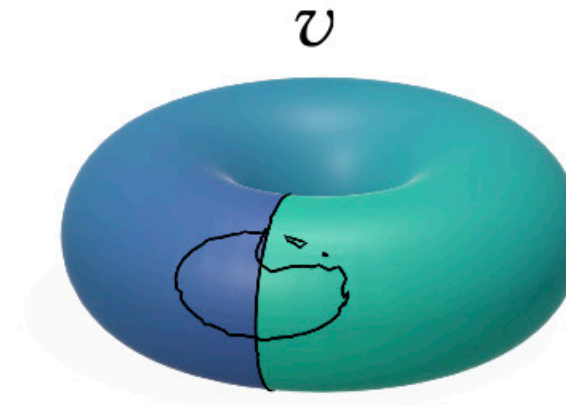
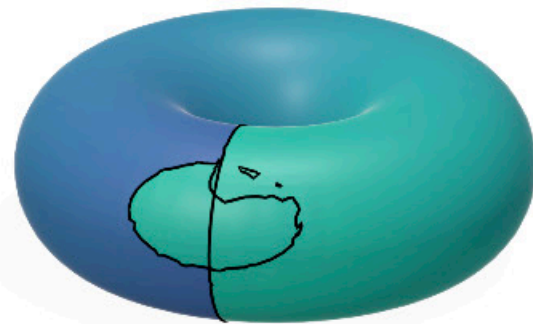
2. Minimize the L^1 norm of jumps across connected components.



Integrate γ locally within each connected component



Dijkstra-completed curve



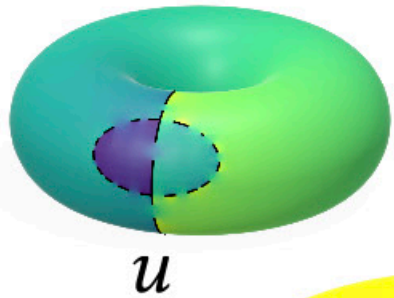
Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .

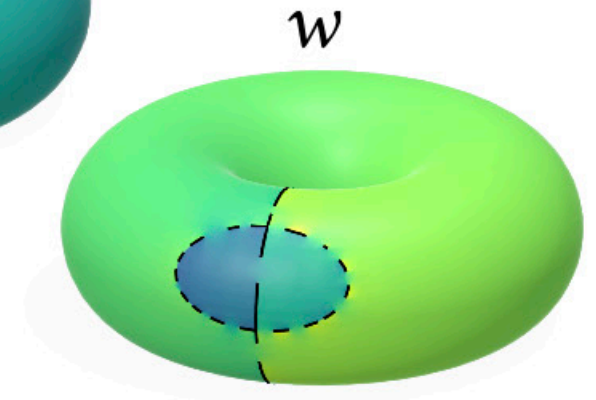
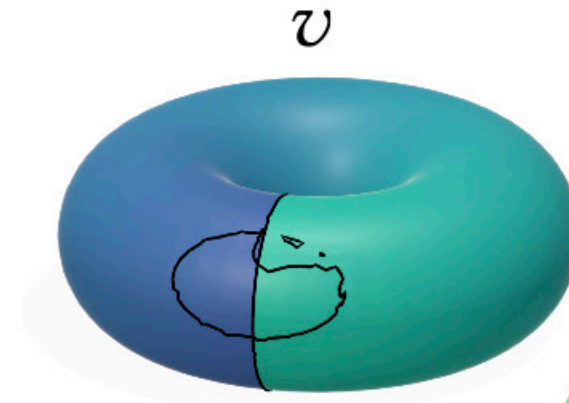
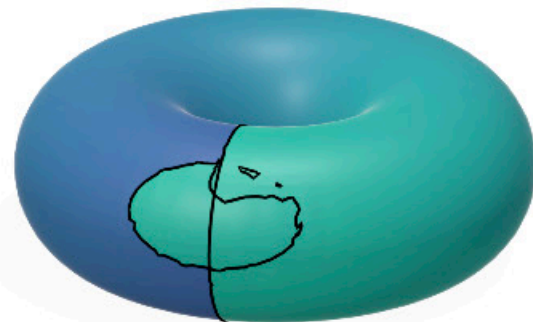
2. Minimize the L^1 norm of jumps across connected components.



Integrate γ locally within each connected component



Dijkstra-completed curve



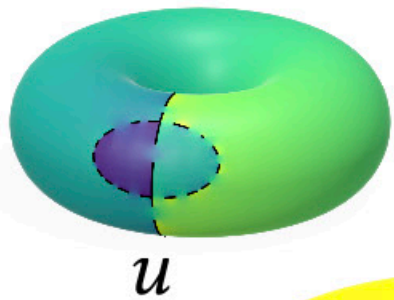
Reduced-size linear program

Current implementation simultaneously optimizes both jump locus *and* jump magnitudes.

Instead:

1. Use a *separate* shortest-path heuristic (Dijkstra) to complete Γ .

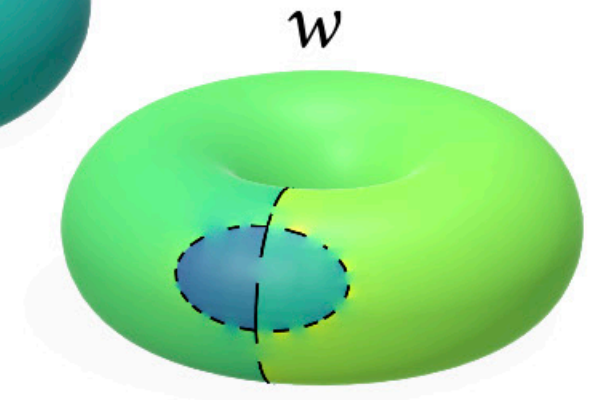
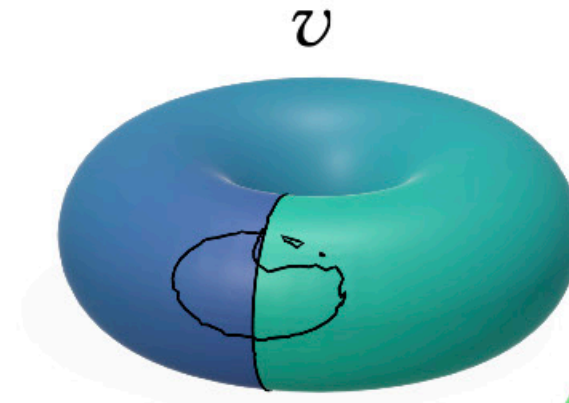
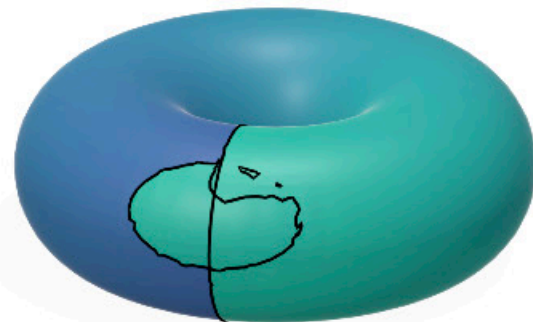
2. Minimize the L^1 norm of jumps across connected components.



Integrate γ locally within each connected component



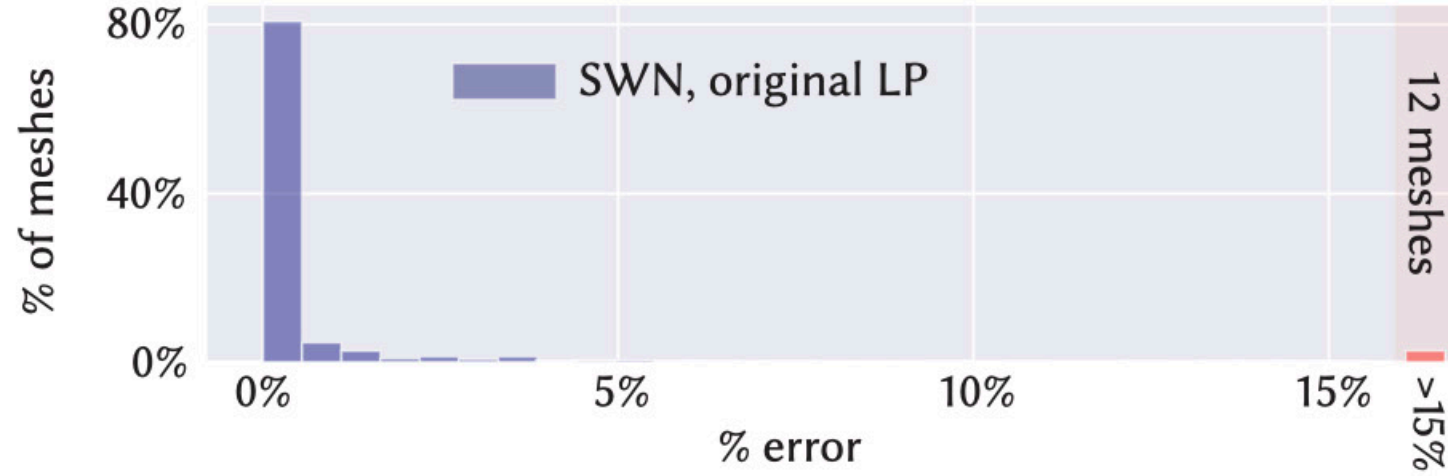
Dijkstra-completed curve



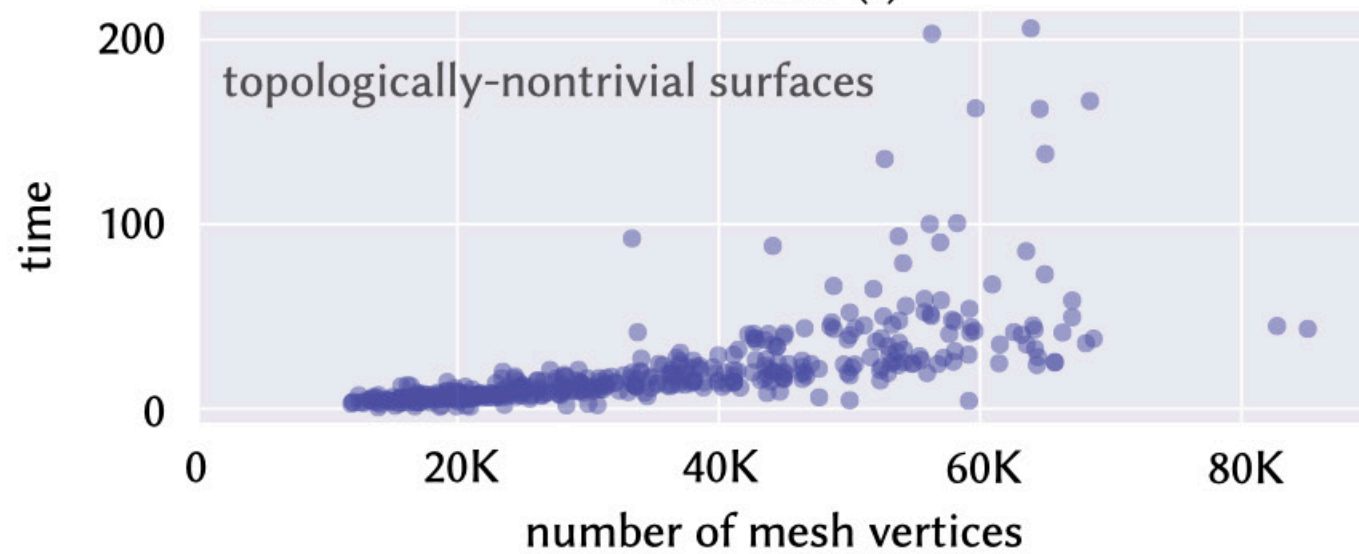
Number of DOFs: $|F|$ \rightarrow just a few connected components!

Approximate solutions, 100x faster

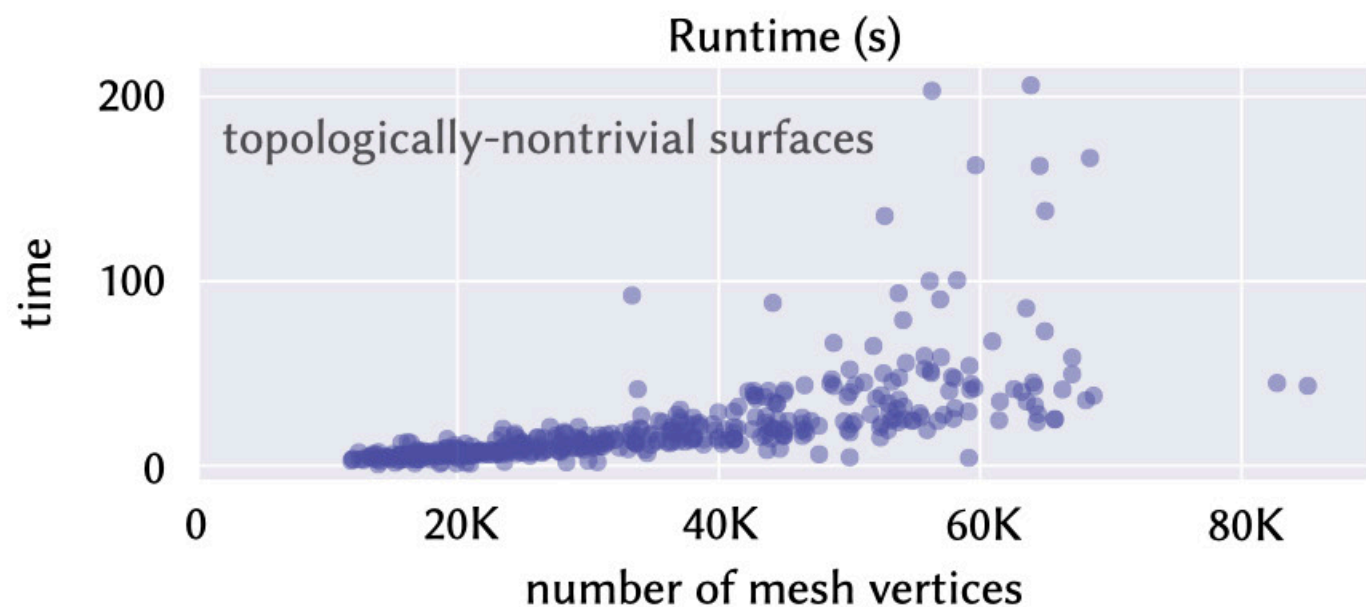
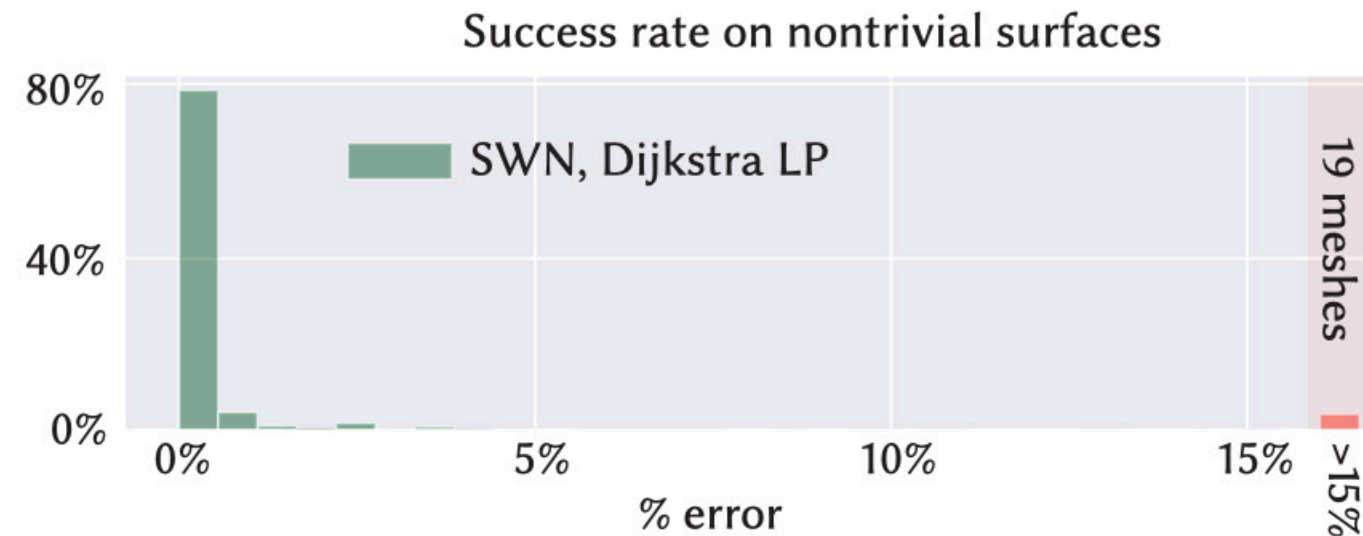
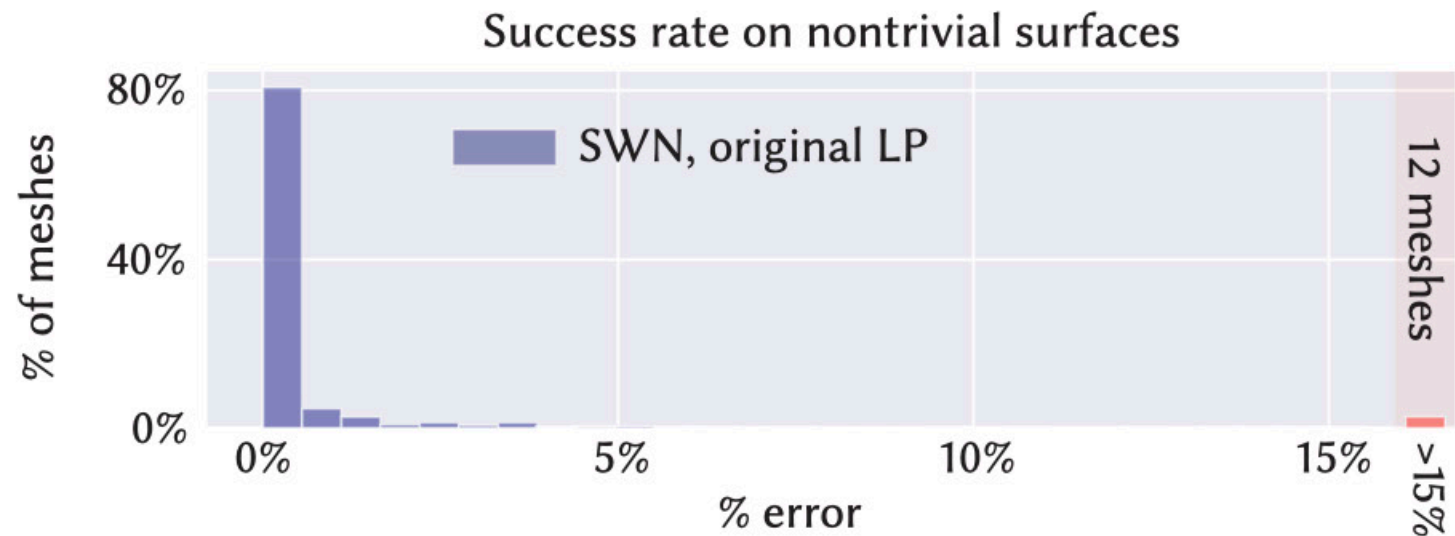
Success rate on nontrivial surfaces



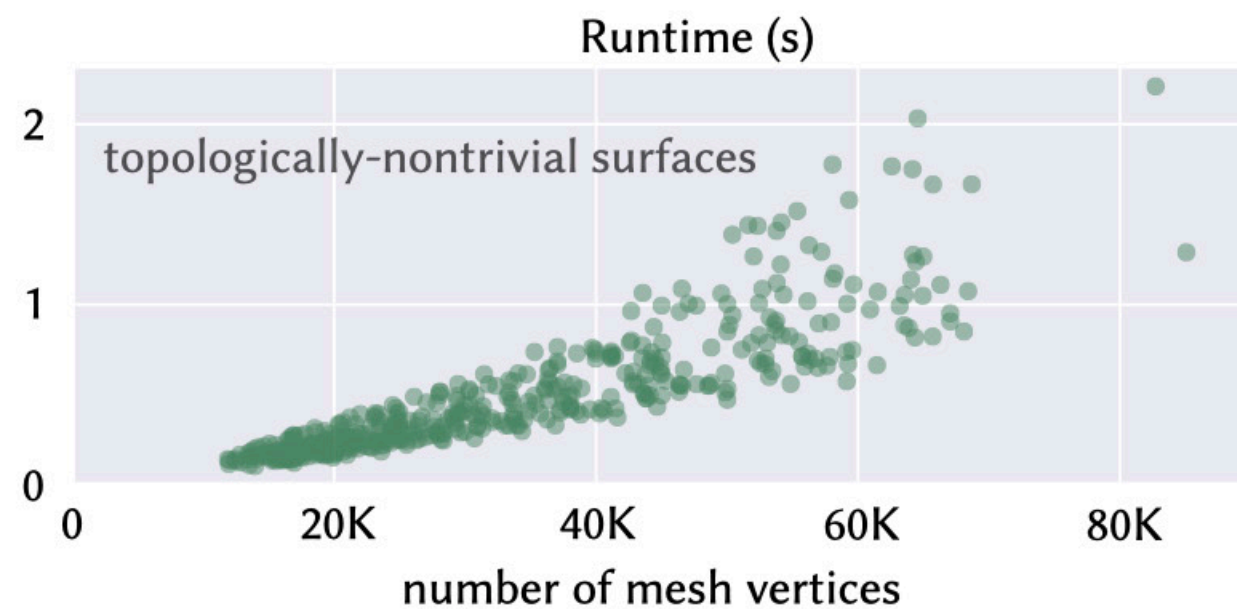
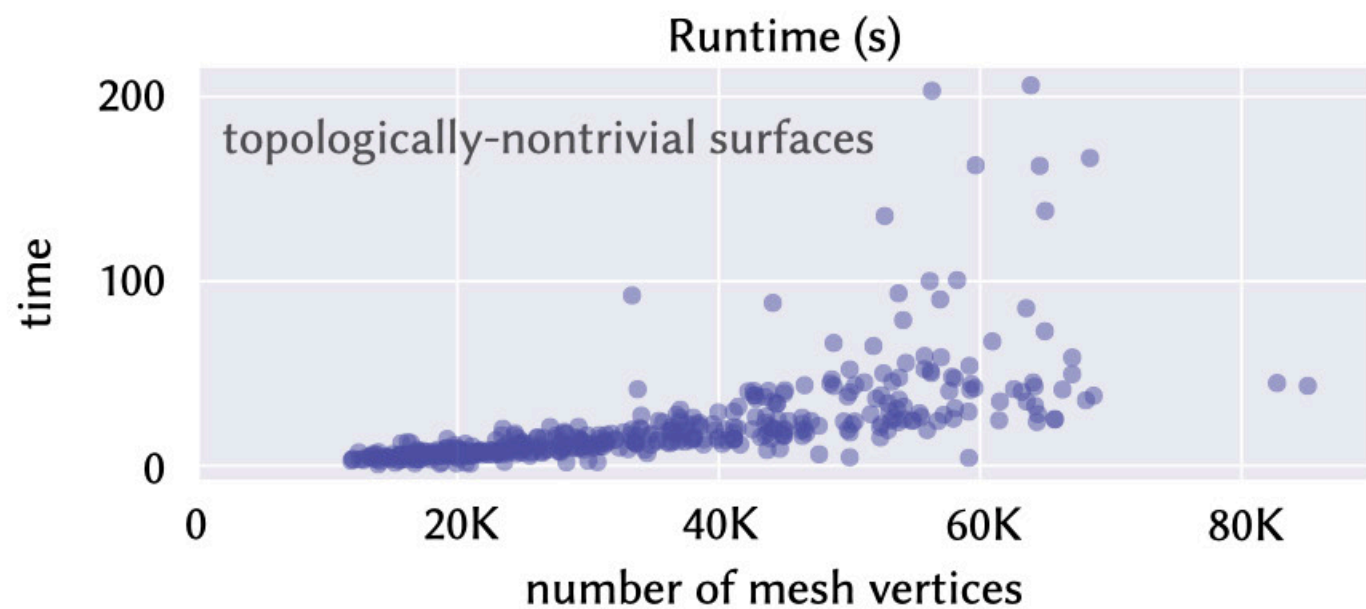
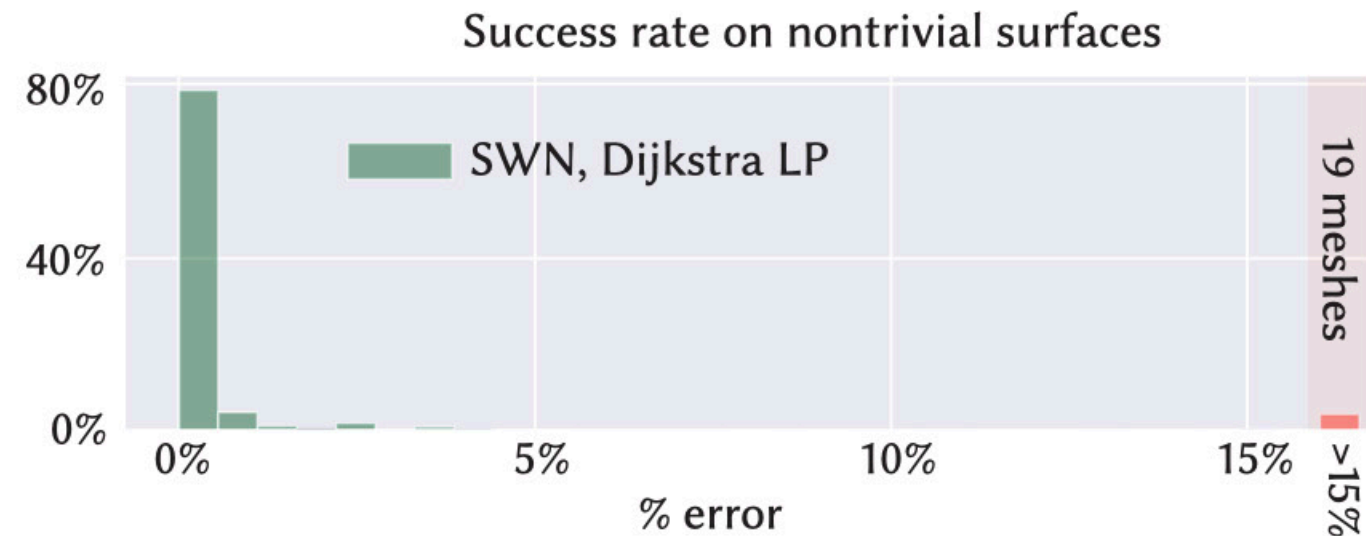
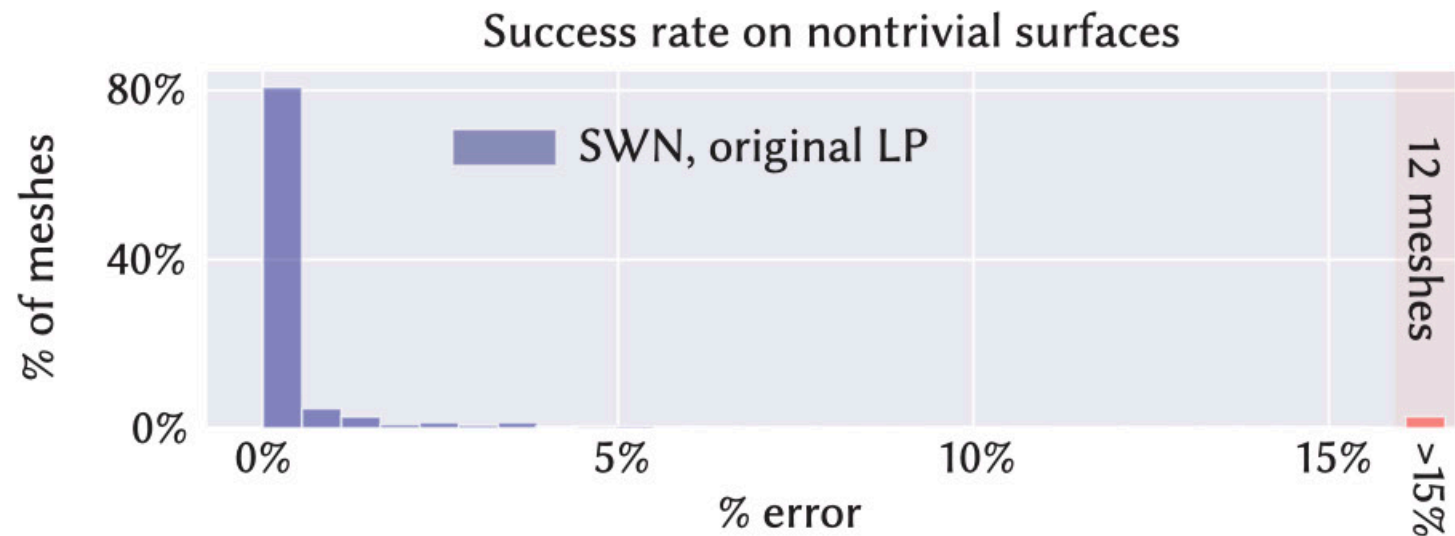
Runtime (s)



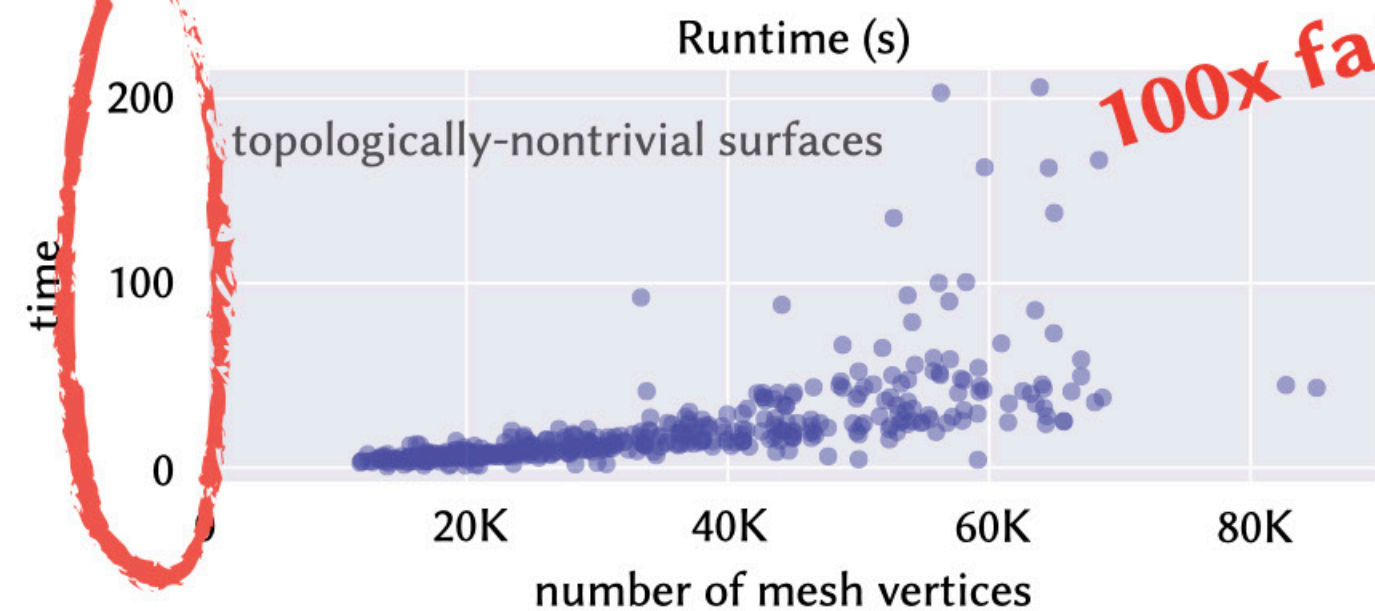
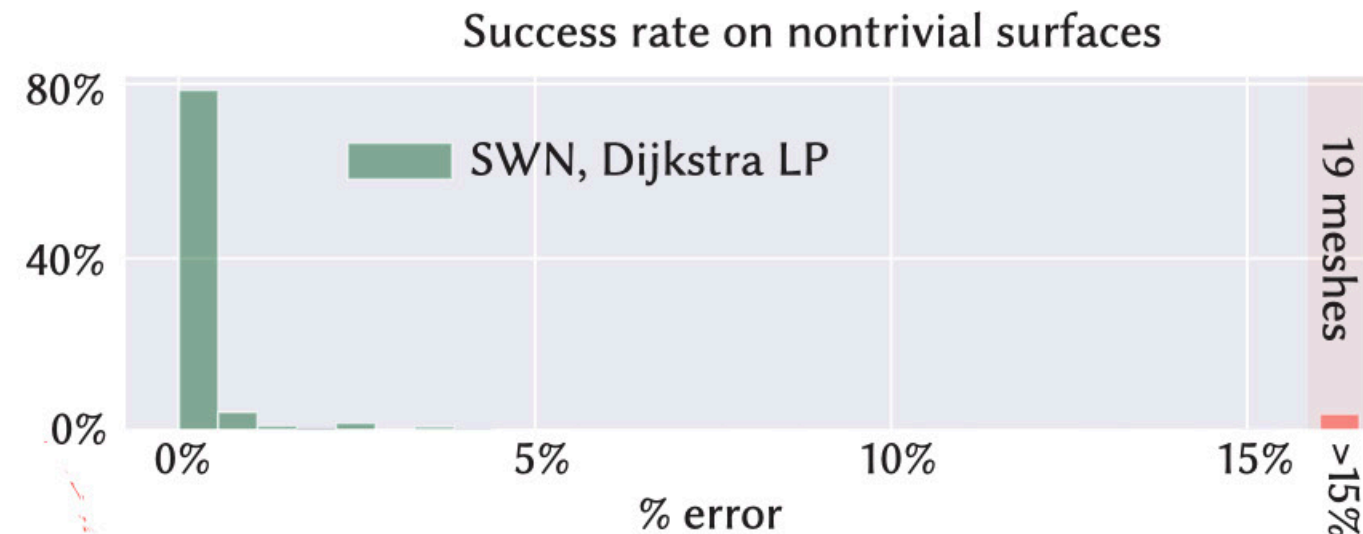
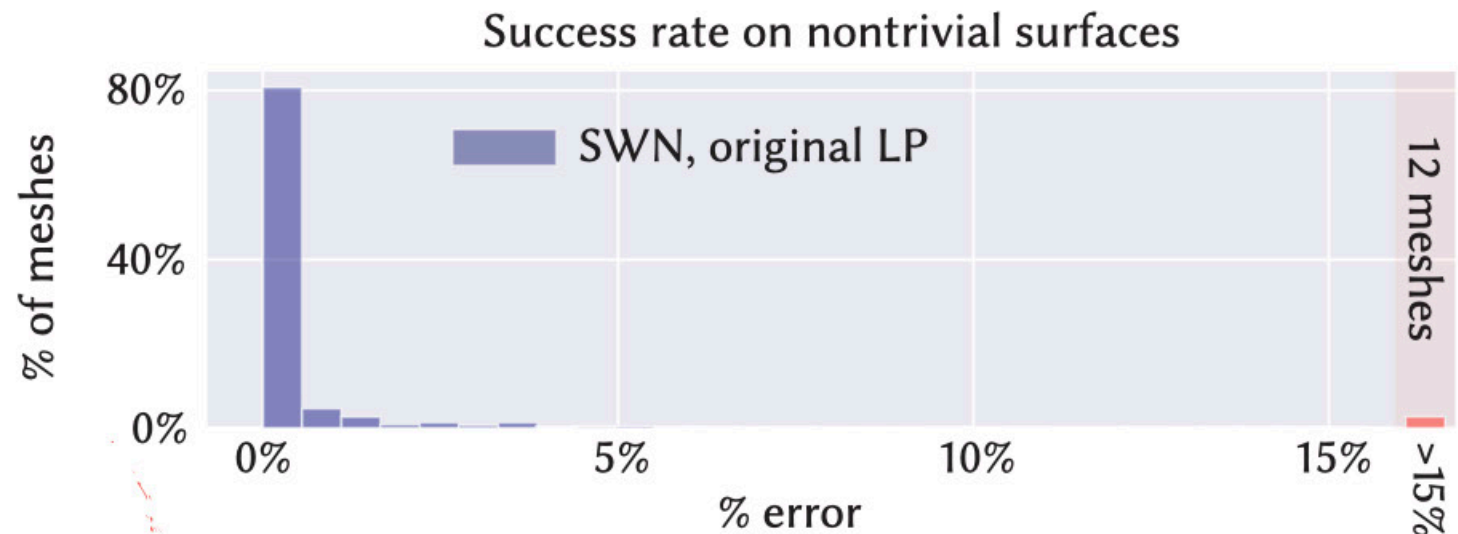
Approximate solutions, 100x faster



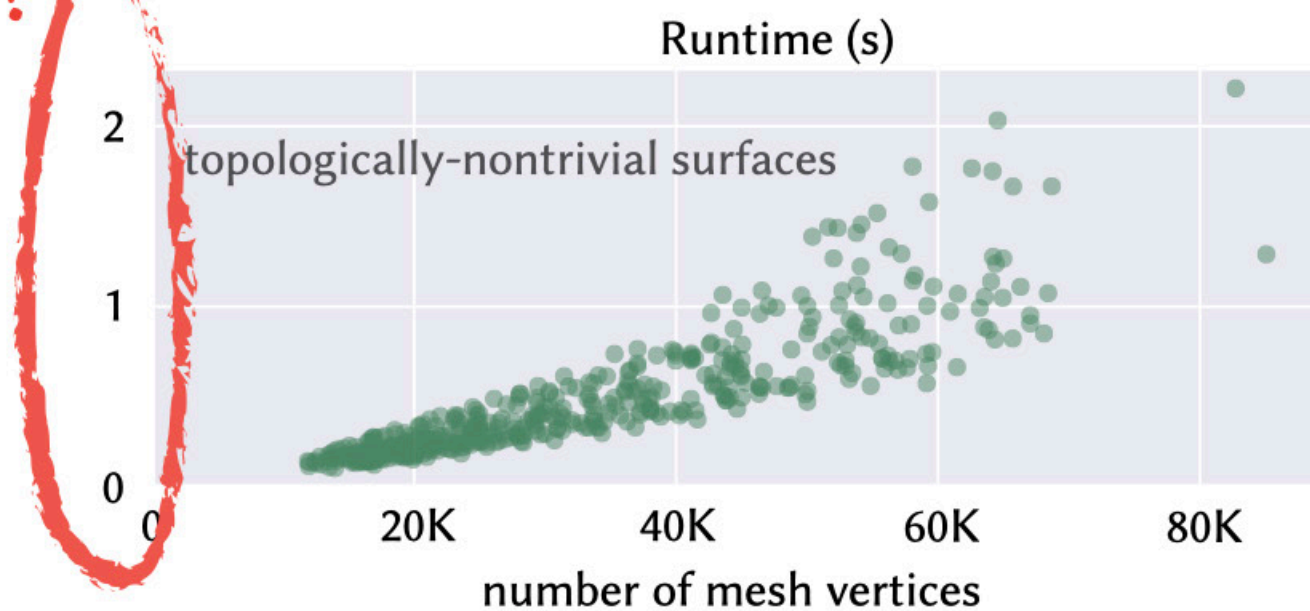
Approximate solutions, 100x faster



Approximate solutions, 100x faster



100x faster!



CONCLUSION

Takeaways

Takeaways

- Classic inside-outside definitions don't work on surfaces!



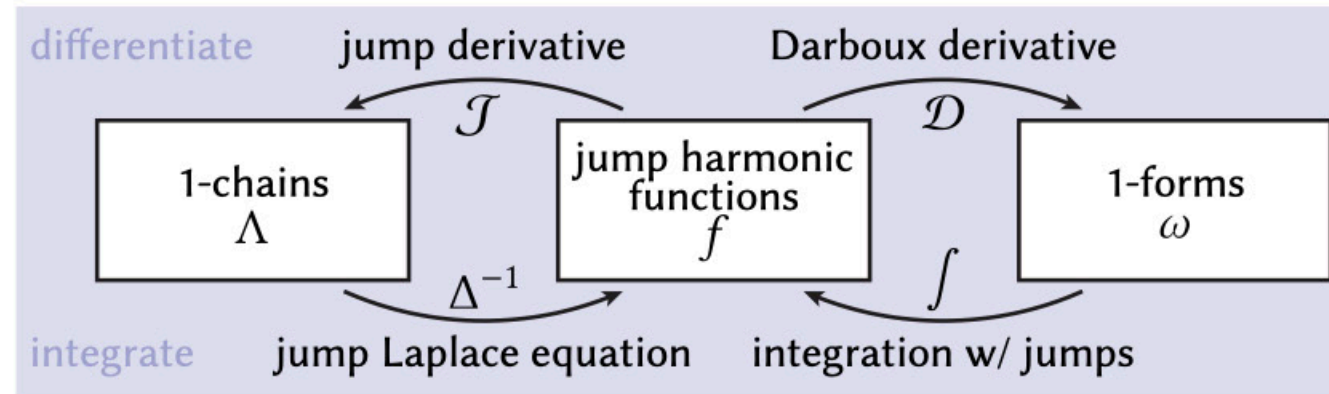
Takeaways

- Classic inside-outside definitions don't work on surfaces!
- Cohomology \rightarrow robust homological geometry processing



Takeaways

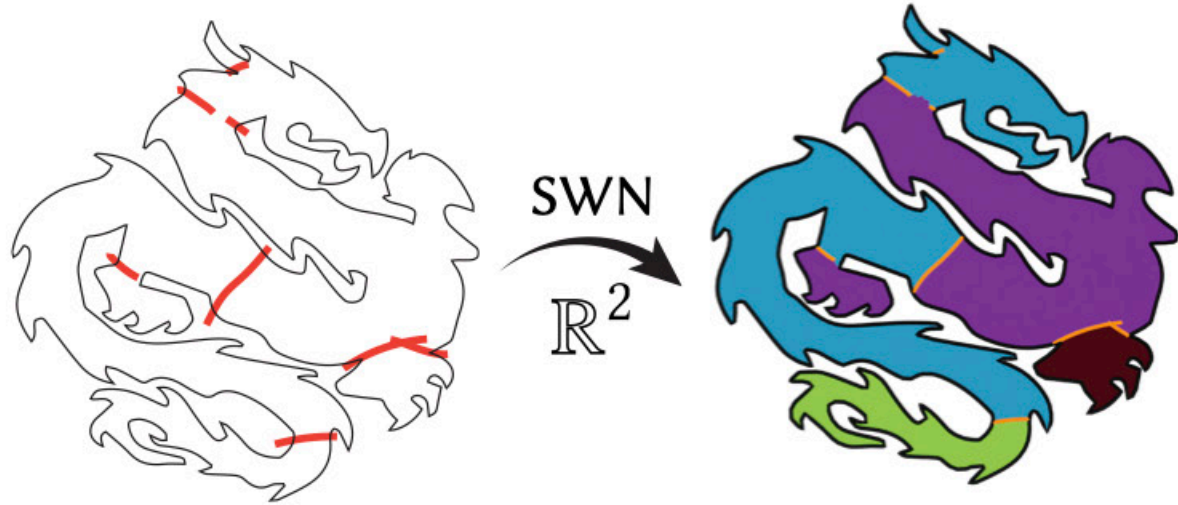
- Classic inside-outside definitions don't work on surfaces!
- Cohomology \rightarrow robust homological geometry processing
- Duality between curves and 1-forms \rightarrow use jump harmonic functions to translate between the two



Fun future directions

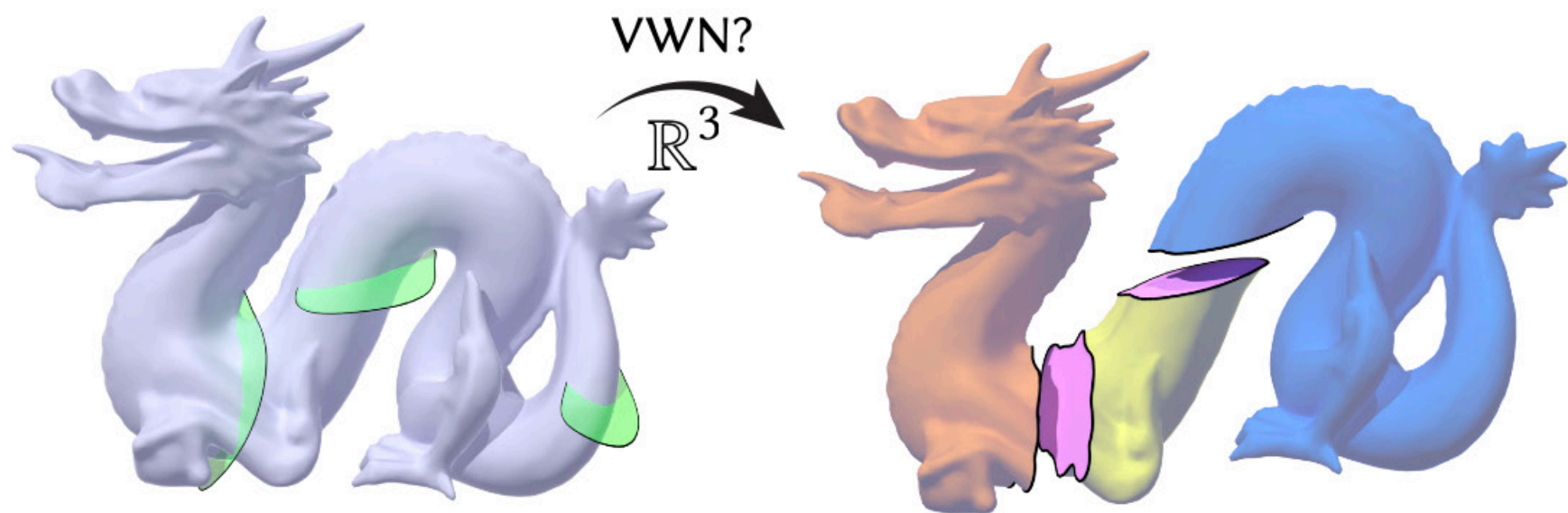
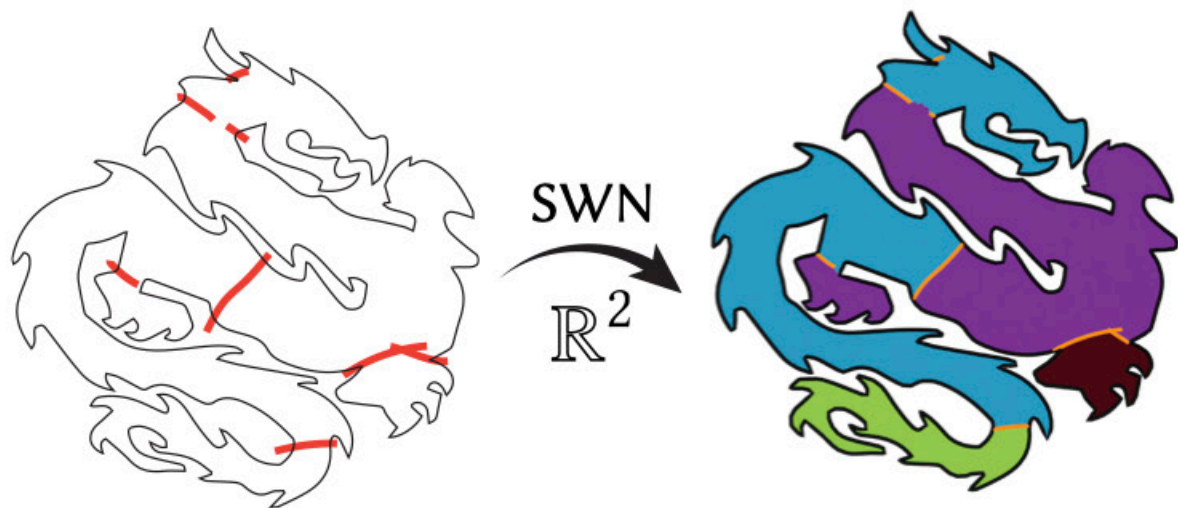
Fun future directions

Subsets of \mathbb{R}^n :



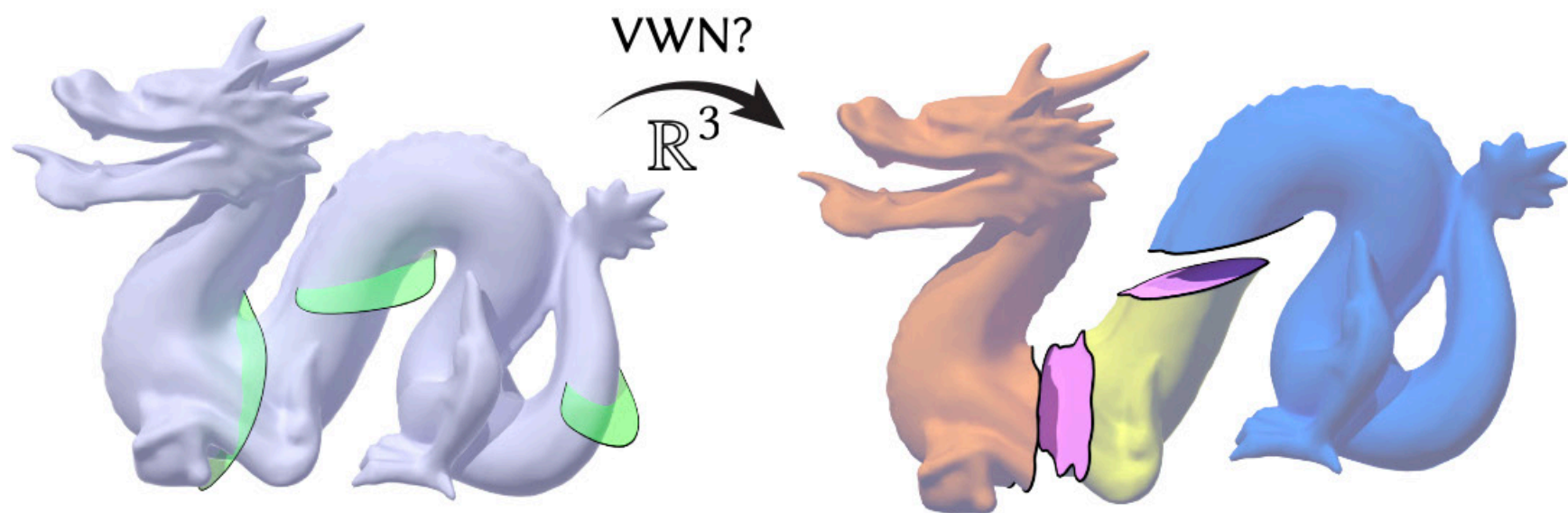
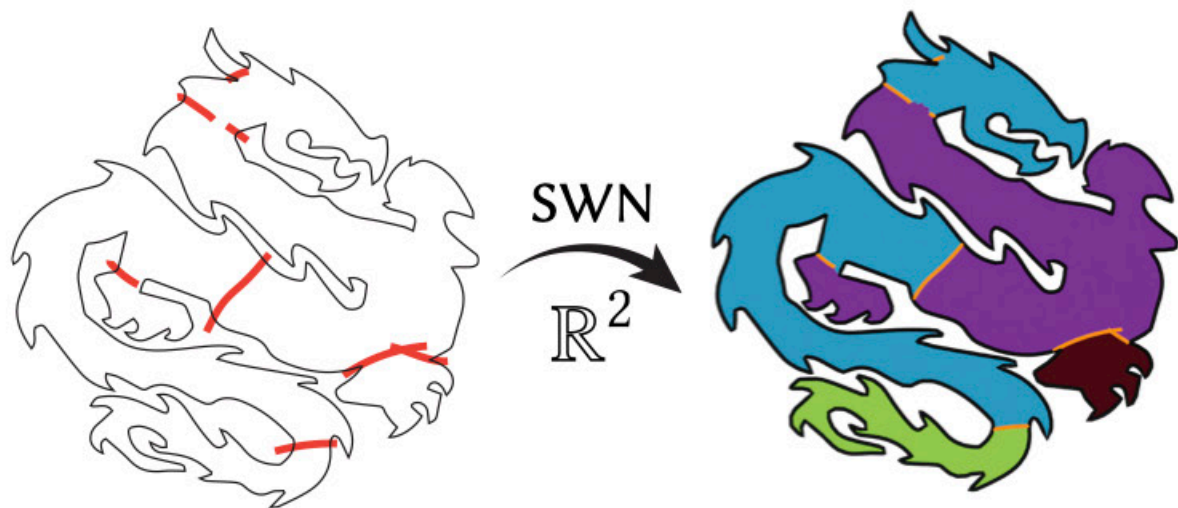
Fun future directions

Subsets of \mathbb{R}^n :

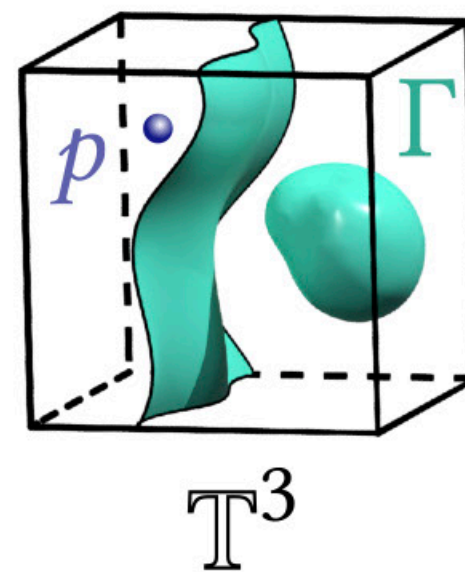


Fun future directions

Subsets of \mathbb{R}^n :



Extension of SWN to higher dimensions,
e.g. periodic domains in 3D.



Winding numbers are everywhere!

Many mathematical & physical interpretations — see our supplemental for details!

PERSPECTIVES ON WINDING NUMBERS
Nicole Feng, Mark Gillespie, Keenan Crane

This short note explores the many different ways one can characterize the winding number of a curve Γ around a point p , and why those standard perspectives fail to generalize to curves on surfaces. Ultimately, all perspectives lead back to one of just three analytical descriptions: an integral over the curve Γ , an integral over a circle around the point p , or a particular Laplace equation. On surfaces, however, these formulations have undesirable consequences for curves that do not correspond to region boundaries, helping to motivate the recent *surface winding number* approach of Feng et al. [2023].

NOTATION AND CONVENTIONS

We use $|\cdot|$ and $\langle \cdot, \cdot \rangle$ to denote the standard Euclidean norm and inner product for vectors in \mathbb{R}^2 . We use $J: \mathbb{R}^2 \rightarrow \mathbb{R}^2$, $(x, y) \mapsto (-y, x)$ to denote a quarter turn in the counter-clockwise direction. For any two vectors $u, v \in \mathbb{R}^2$, we define a scalar-valued cross product $u \times v := u_1 v_2 - u_2 v_1$; note that $\langle Jx, x \rangle = u \times v$. For any function $f(t)$ of a single parameter t , we let $f'(t) := \frac{d}{dt} f(t)$.

Throughout we consider compact curves Γ as a smooth surface M , possibly with boundary ∂M ; an important special case is the Euclidean plane $M = \mathbb{R}^2$. We use S^1 to denote the circle, which serves as the domain for a single closed loop. More generally, we use I to denote the domain of Γ , which may be an open interval, a closed loop, or a larger collection of loops and intervals. We use $w_\Gamma(p)$ to denote the winding number of a curve (or collection of curves) Γ around a point p ; when Γ is not closed, this function also describes the signed solid angle.

We use Δ to denote the negative-semidefinite Laplace-Beltrami operator on M , which locally behaves like the ordinary Laplace operator $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. A function $u: M \rightarrow \mathbb{R}$ is harmonic if it is in the kernel of Δ (Laplacian), i.e., $\Delta u = 0$.

1. TOPOLOGICAL DEGREE

The basic idea of the winding number is that it captures how many times a curve Γ “winds” around a given point p . In particular, for a single closed planar loop $\Gamma: S^1 \rightarrow \mathbb{R}^2$, consider the covering map

$$\varphi: S^1 \rightarrow S^1, t \mapsto \frac{\Gamma(t) - p}{|\Gamma(t) - p|} \quad (1)$$

The winding number $w_\Gamma(p)$ can then be defined as the degree of φ , i.e. the number of times φ covers the circle S^1 , taking orientation into account. For instance, if φ goes once around the circle in a counter-clockwise direction we get a winding number $+1$; in the clockwise direction we get -1 , and if φ goes around the circle multiple times we get a winding number of magnitude greater than one (Figure 1). The winding number is also given by the total signed length of the image of S^1 under φ , divided by the circumference of the circle. Since $\varphi(t)$ is always a point on the unit circle, we can express the infinitesimal signed length as $\langle \varphi'(t), \varphi(t) \rangle$. Let ds denote the length of the tangent vector $\varphi'(t)$ along the counter-clockwise direction $\varphi(t)$ tangent to the circle (Figure 2, left). The total signed length is then

2 • Feng, Gillespie, Crane

Fig. 1. For curves Γ in the plane, the winding number function $w_\Gamma(p)$ gives the number of times the curve Γ wraps around any given point p .

Fig. 2. Left: The solid angle function (a.k.a. the generalized winding number) is the total signed length of the projection of Γ onto a circle around p (Figure 1, left). Unlike physical solid angle, however, the signed subtended angle is oriented with multiplicity, and will be negative whenever Γ and S^1 are oppositely oriented. One can interpret the signed length of p as the total signed angle subtended by the curve Γ over a small circle around p . This idea extends naturally to open curves, in which case the subtended angle is the fraction of the “sky” covered by Γ for an observer standing at p (Figure 3, left). Unlike physical solid angle, however, the signed subtended angle is oriented with multiplicity, and will be negative whenever Γ and S^1 are oppositely oriented. On a surface, one might be inclined to compute the subtended angle via the logarithmic map. At any point $p \in M$, the logarithmic map \log_p gives the direction u and shorter distance $|u|$ we must walk along a straight path (i.e. geodesic) to reach q . Letting $R(p)$ be the angle of the vector $\log_p(q)$, we could then try integrating the quantity dR to obtain a notion of winding number. The problem, however, is that there are points at which there is not a unique shortest path to points q on the curve Γ . As one point q crosses through this so-called cut locus, then the integral $\int dR$ may jump discontinuously, as pictured in Figure 3. Moreover, the exponential map may not be surjective on domains with boundary, hence the log map may not be well-defined for some points on our curve.

3. RAY INTERSECTIONS

Alternatively, one can interpret the degree of the map φ as the number of points along Γ which get mapped to a generic point on the circle (counted again with multiplicity). The number of points which φ maps onto a unit vector v is precisely the number of signed intersections $\varphi(t)$ between Γ and a ray leaving p in the direction v , i.e., $+1$ if the ray has a positive dot product with the unit normal n of Γ , and -1 otherwise. Hence, one can evaluate the winding number of a point p by shooting a random ray from p and counting the number of intersections with Γ (Figure 4, left).

On a surface, the natural analogue of shooting a ray is to evaluate the exponential map $\exp_p(v)$, which traces out a geodesic curve starting at p in the direction v for time t . Unlike the plane, however, a geodesic may intersect a closed curve Γ infinitely many times, artificially truncating it to finite length yields an arbitrary answer (Figure 4, center); moreover, the number of intersections may also change completely with the ray direction v (Figure 4, right).

Fig. 3. Left: A naive approach to assigning winding numbers to regions does not work on surfaces. For instance, even if we always assign “0” to the larger region, the subsequent labeling may depend arbitrarily on the order in which we visit neighboring regions.

3 • Perspectives on Winding Numbers (Technical Note)

Fig. 4. Left: In the plane, the winding number of a closed curve Γ can be found by counting the number of signed intersections with a generic ray: here, $w_\Gamma(p) = +1$ if v and $w_\Gamma(p) = -1$ if v is π . Center, right: on a surface, a geodesic ray may intersect a closed curve infinitely many times, or change completely depending on the initial direction v .

Fig. 5. Top: In the plane, the winding number function can be viewed as the electric potential as the number of dipoles it goes to infinity—shown here for an open and closed curve, where normals a determine dipole moments. Bottom: on surfaces this same harmonic function is well-defined, but may not provide a meaningful notion of inside/outside even for closed loops.

4. ELECTROSTATICS

A more direct connection between winding numbers and electric fields is given by Gauss’ law, which states that the flux of the electric field through a closed surface Γ is given—up to a constant—by the enclosed charge Q :

$$\int_{\Gamma} \mathbf{E} \cdot d\mathbf{s} = \frac{Q}{\epsilon_0} \quad (6)$$

If we place a single point charge at p , then the resulting electric flux through Γ is precisely the winding number of Γ around p . The electric field \mathbf{E} induced by a point charge can be written as the gradient of an electric potential ϕ , which can be found as the solution to a Poisson equation. One can extend this procedure to surfaces, solving a Poisson equation for the electric potential of a point charge at p , taking its gradient to obtain the electric field \mathbf{E} , and then computing the flux through Γ (center). This gives the same solution as the jump equation in Section 5, but is less attractive computationally because one must solve a PDE over per evaluation point, rather than once per curve.

5. HARMONIC FUNCTIONS

As noted in Feng et al. [2023, Section 1] and in Section 7, the winding number function in the plane is also a particular harmonic function—for a simple curve Γ in the plane it is a solution to a Laplace equation with jumps, namely,

$$\begin{aligned} \Delta u &= 0, & \text{on } \mathbb{R}^2 \setminus \Gamma, \\ u^+ - u^- &= 1, & \text{on } \Gamma, \\ \Delta u^* / \Delta u &= \Delta u^* / \Delta u, & \text{on } \Gamma. \end{aligned} \quad (10)$$

The boundary conditions for this Laplace equation are somewhat unusual, rather than prescribing function values or normal derivatives along Γ (i.e. Dirichlet or Neumann conditions, resp.), we have jump boundary conditions, which say that the two solution values u^\pm on either side of the curve must differ by one, and that the normal derivative must be equal on both sides of the curve (Dziukinski [2001] gives a more formal treatment). Equation 4 is readily solved for curves on surfaces, and serves as the starting point for the formulation in [Feng et al. 2023]. However, it does not immediately resolve the question of how to define winding numbers on surfaces, since even for closed curves the solution a may not be a piecewise integer function (Figure 6, bottom right shows one example).

The PDE perspective can be connected to the standard definition of winding numbers via the idea of a double layer potential. Conceptually, we imagine that a collection of equal-magnitude positive and negative electric charges are lined up along the curve Γ . Each positive/negative particle pair has an associated dipole potential, and as we pack more and more charges along Γ (as in Figure 6, top), the sum of these potentials converges to a harmonic function with a constant jump across Γ (see [Bershad et al. 1984, pp. 56–58] and [Hsiao and Wroldland 2008, Ch. 1] for more formal discussion). More

4 • Feng, Gillespie, Crane

Fig. 6. Top: The winding number function $w_\Gamma(p)$ is essentially the “shadow” of a other complex function. For instance, the winding number integral in Equation 3 is the real part of the complex integral

$$w_\Gamma(p) = \frac{1}{2\pi i} \int_{\Gamma} \frac{1}{z - p} dz, \quad (7)$$

where i is the imaginary unit, and we view Γ as a complex-valued curve $\Gamma: I \rightarrow \mathbb{C}$. By exponentiating, we may obtain a complex function $f(p) = \exp(2\pi i w_\Gamma(p))$ whose argument—i.e. angle from the origin—is given by the real part of $w_\Gamma(p)$, and whose magnitude is determined by the imaginary part of $w_\Gamma(p)$. When Γ is an open curve from a to b , we can write $f(p)$ in closed form:

$$f(p) = \frac{z - b}{z - a} \quad (8)$$

as shown in Figure 7. The function f depends only on Γ ’s endpoints, oblivious to the location of the curve itself, since angle-valued functions ignore the integer jumps across Γ . In general, f has zeros

at positive endpoints of Γ and poles at negative endpoints. Hence, $w_\Gamma = \frac{1}{2\pi i} \log f$ has logarithmic singularities at all endpoints of Γ , where it locally looks like the function $\log(z)$.

An analogous procedure can be performed on orientable surfaces, constructing a complex function associated with Γ whose argument is given by w_Γ . While the complex perspective does not resolve the fundamental ambiguity of defining winding number on surfaces, the description of the logarithmic singularities around the endpoints of Γ is helpful when discretizing and interpolating jump harmonic functions [Feng et al. 2023, Section 2.3.2].

Fig. 7. Top: The winding number function $w_\Gamma(p)$ is essentially the “shadow” of a other complex function. For instance, the winding number integral in Equation 3 is the real part of the complex integral

Perspectives on Winding Numbers
Nicole Feng, Mark Gillespie, Keenan Crane