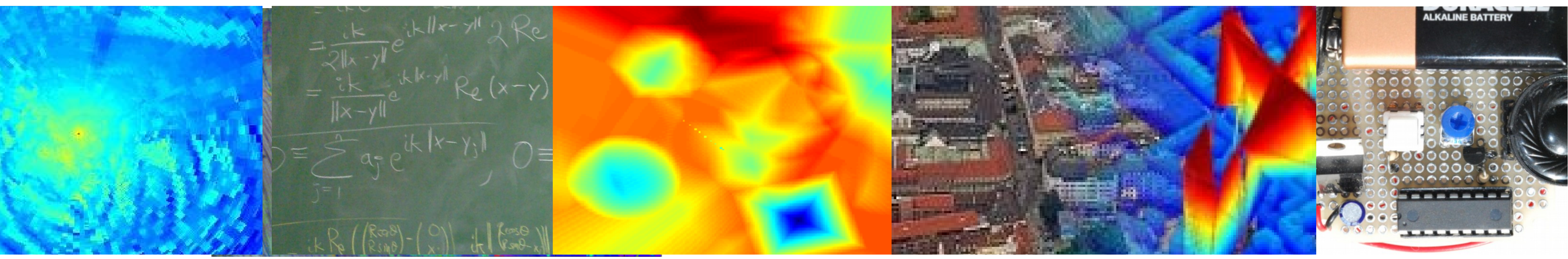
# Topological Filters
## A Toolbox for Processing Dynamic Signals




Michael Robinson

# Acknowledgements


- Collaborators:
  - Cliff Joslyn, Katy Nowak, Brenda Praggastis, Emilie Purvine (PNNL)
  - Chris Capraro, Grant Clarke, Janelle Henrich (SRC)
  - Jason Summers, Charlie Gaumond (ARiA)
  - J Smart, Dave Bridgeland (Georgetown)
- Students:
  - Eyerusalem Abebe
  - Olivia Chen
  - Philip Coyle
  - Brian DiZio
  - Jen Dumiak
  - Samara Fantie
  - Sean Fennell
  - Robby Green
  - Noah Kim
  - Fangfei Lan
  - Metin Toksoz-Exley
  - Jackson Williams
  - Greg Young
- Recent funding: DARPA, ONR, AFRL

# Key points

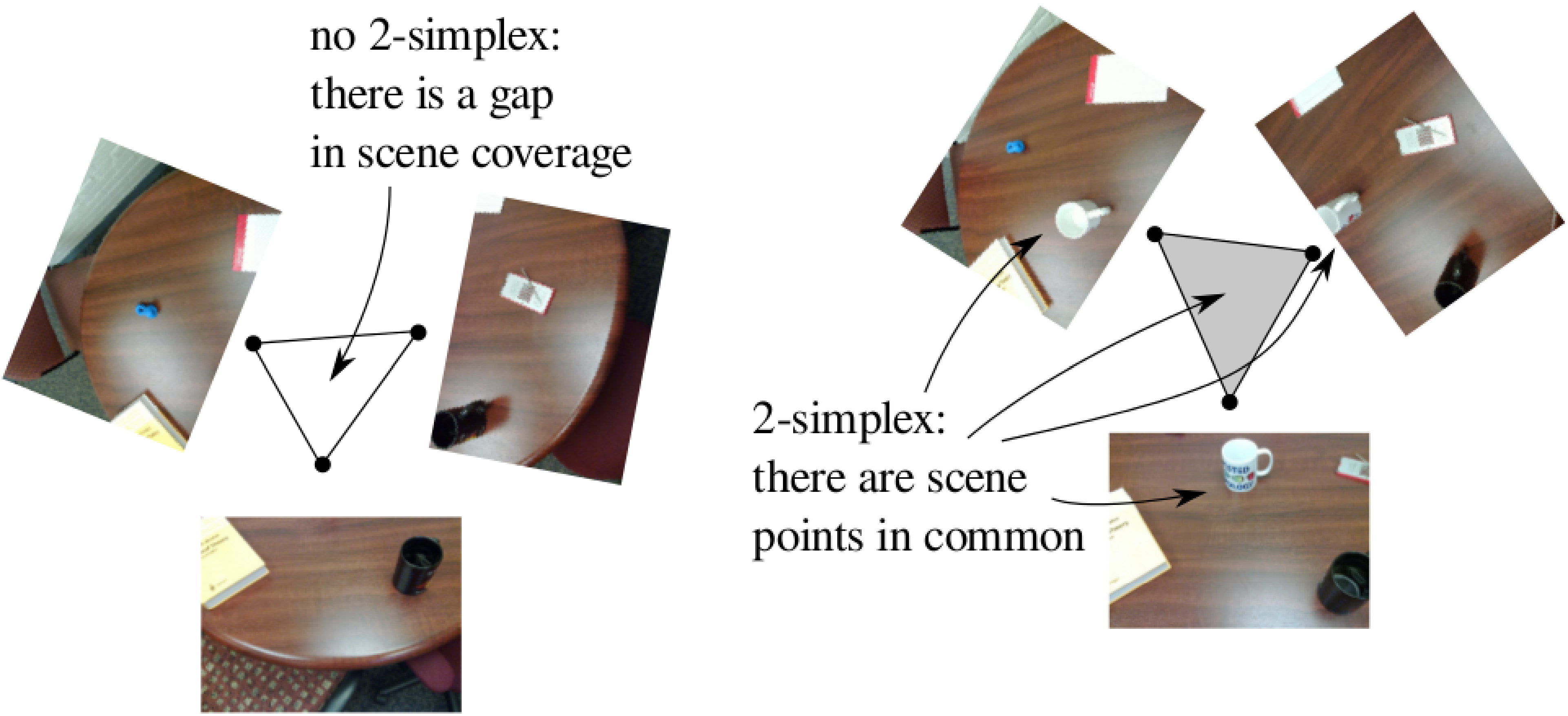
- Systems can be **encoded** as *sheaves*
- Datasets are *assignments* to a sheaf model of a system
- *Consistency radius* **measures compatibility** between system and dataset
  - *Global sections* have zero consistency radius
  - *Data fusion* **minimizes** consistency radius
- Filters **transform** global sections via pairs of *sheaf morphisms*

# What is a sheaf?

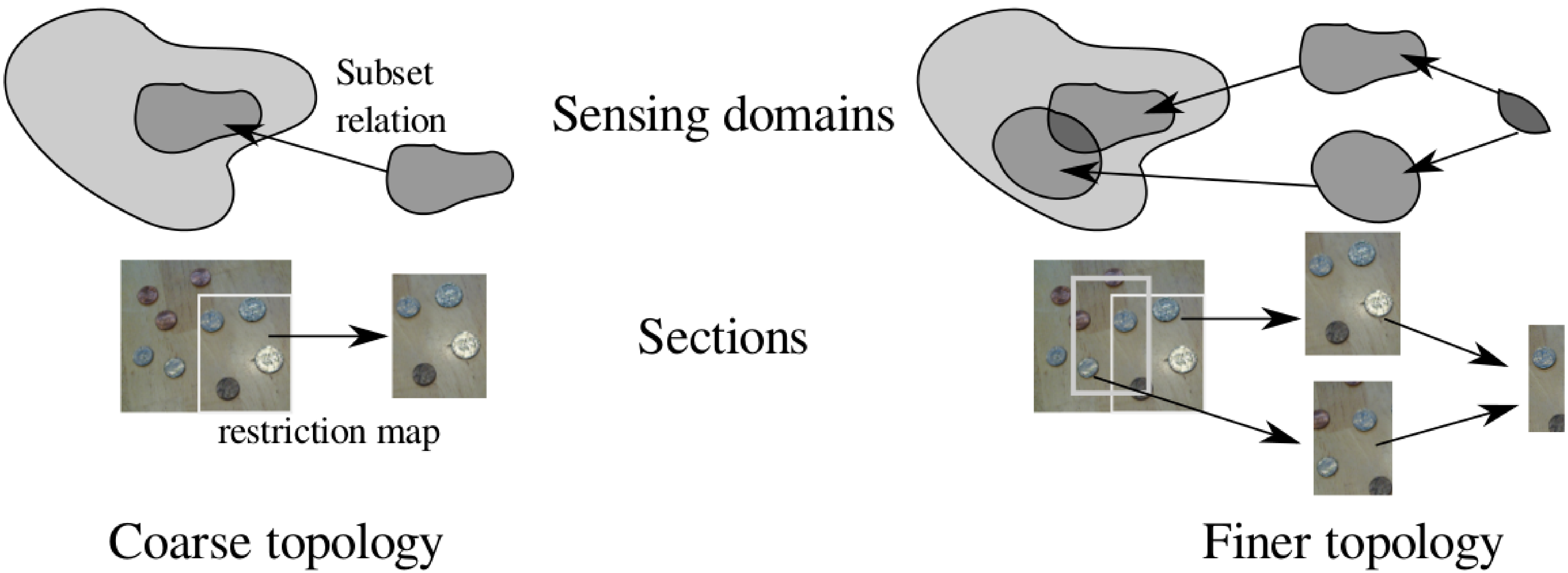
A *sheaf* of _______________ on a ________________

(data type) (topological space)

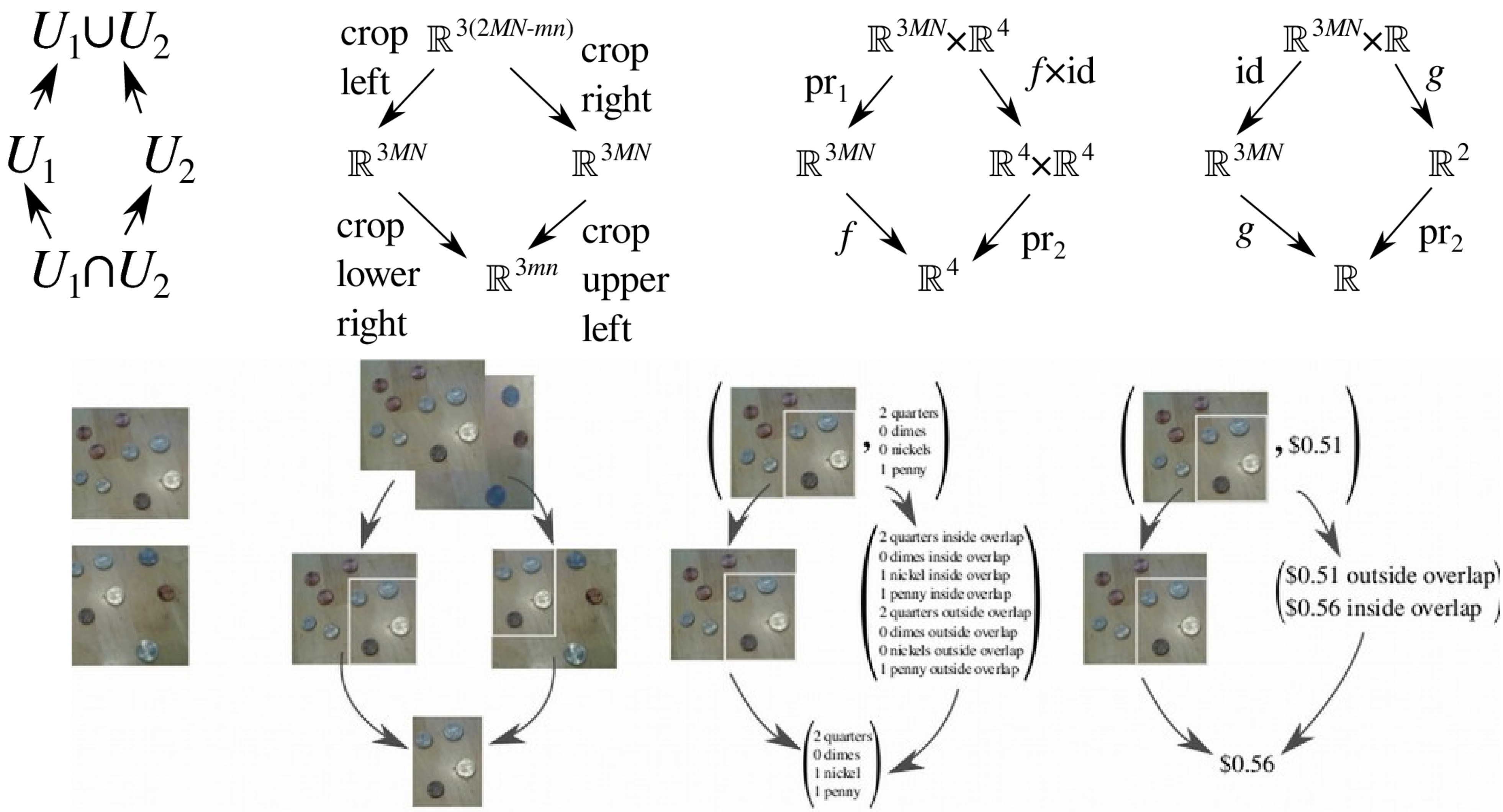
# Overlap constructs topology



# Changing overlaps changes the topology



# Sheaves are about consistency

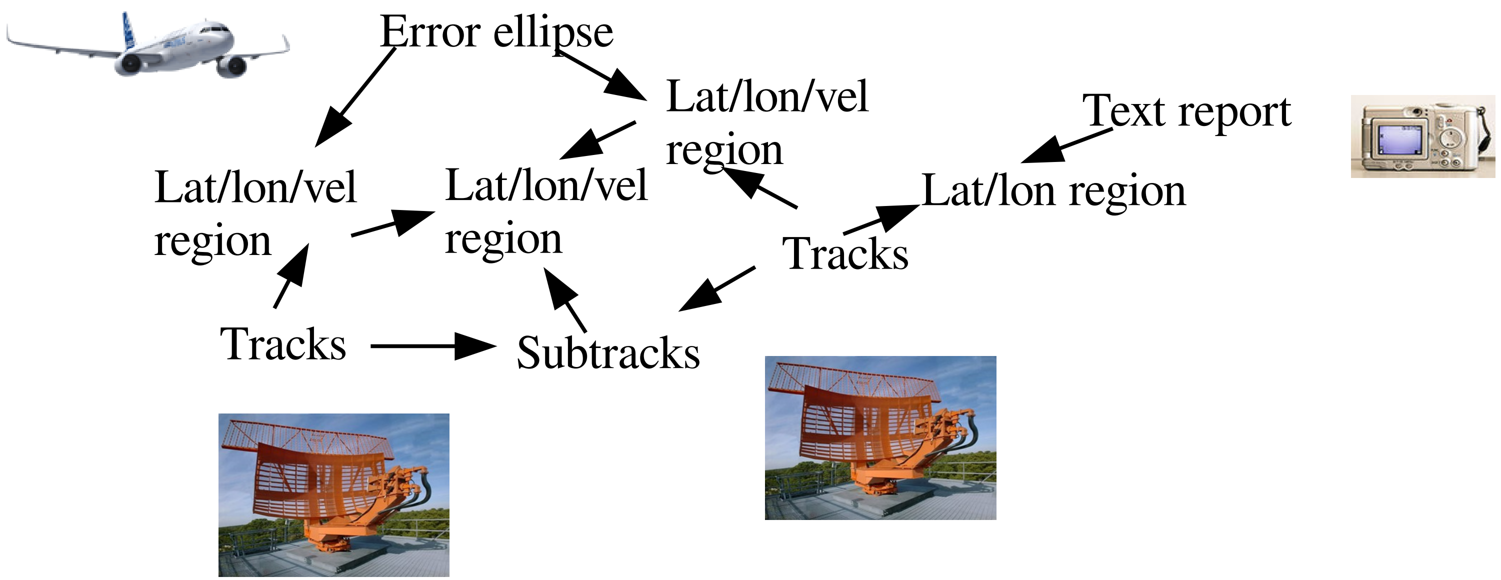



Non-numeric data types of varying complexity can certainly be supported!

Michael Robinson

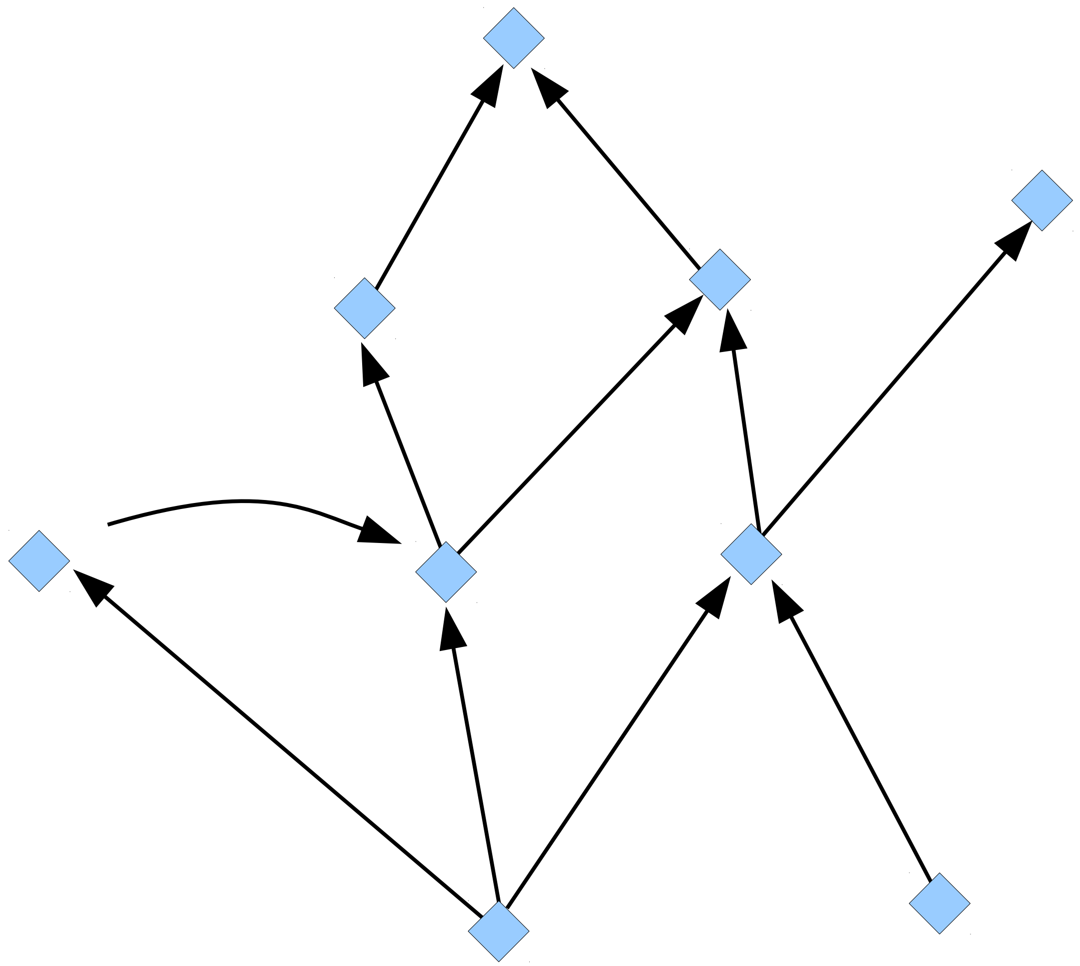
# Finite topologies from partial orders

- *Partial orders* describe the relationships between observations in a system… order relations correspond to (differential) operators



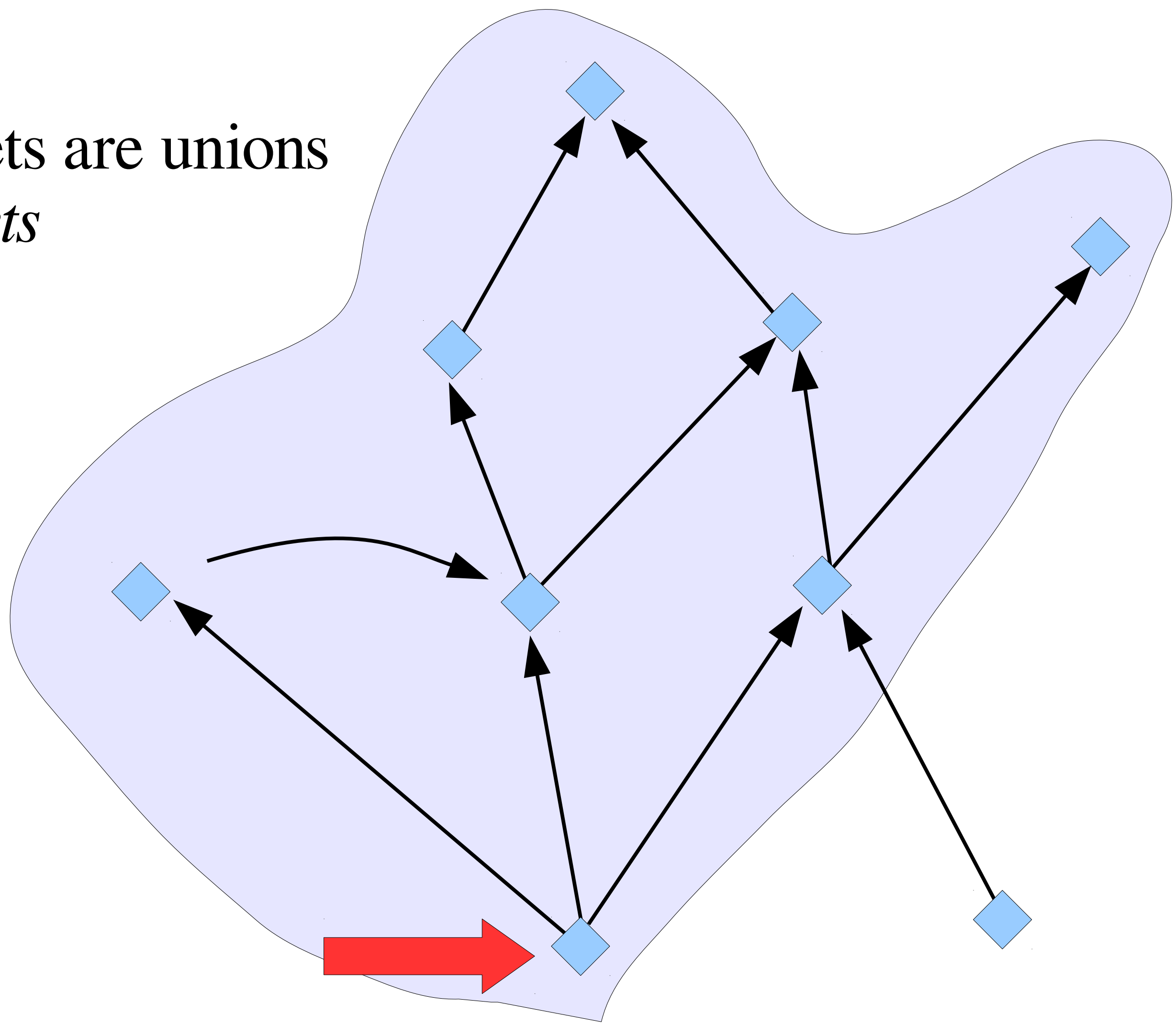

- Every partial order has a natural topology, the *Alexandroff topology*
  - *Presheaves* and *sheaves* “are the same thing” in this topology, since the gluing axiom is satisfied trivially
  - Commutativity is the only actual constraint on a sheaf diagrams

# Topologizing a partial order



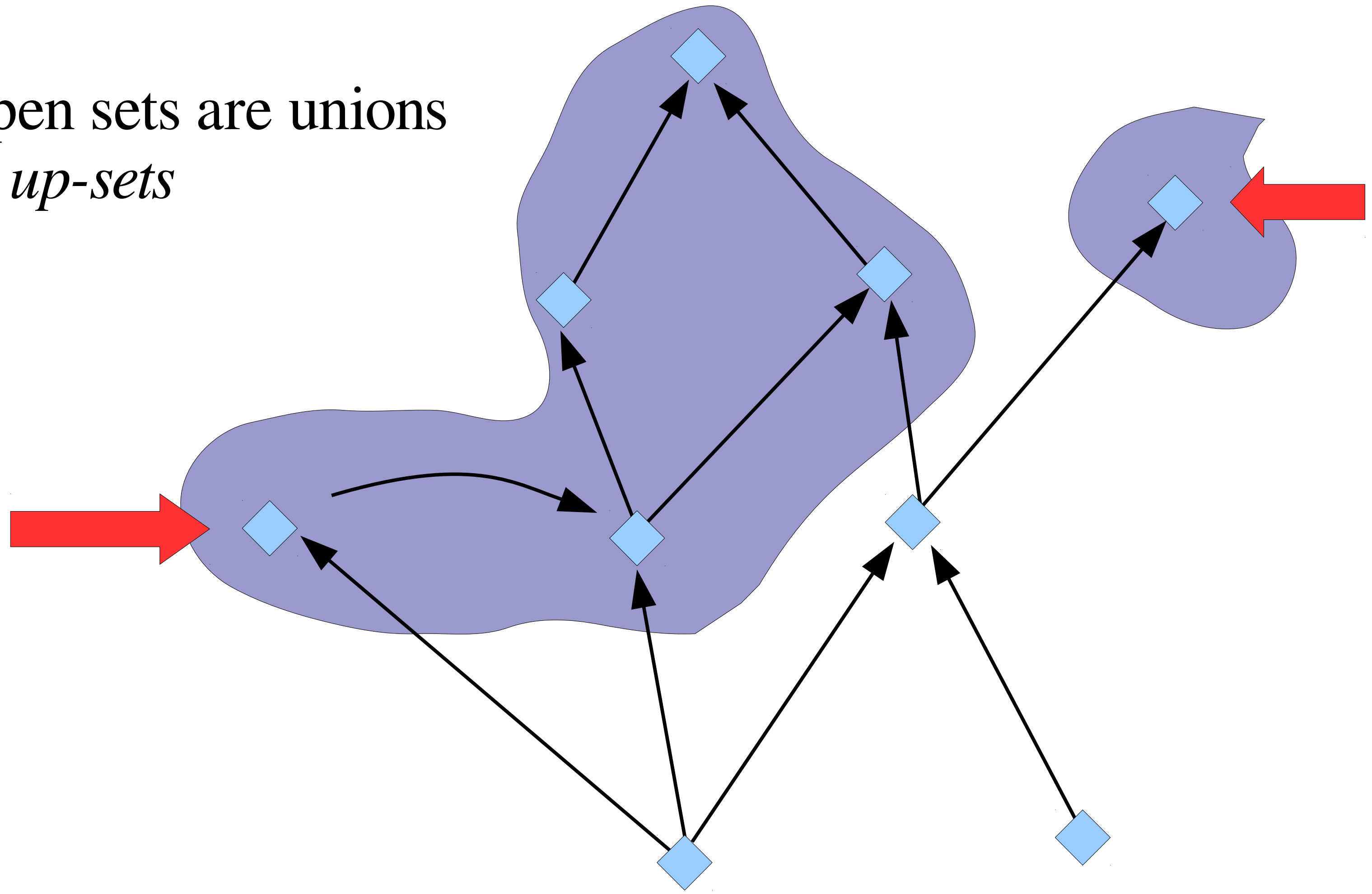
# Topologizing a partial order

Open sets are unions of *up-sets*



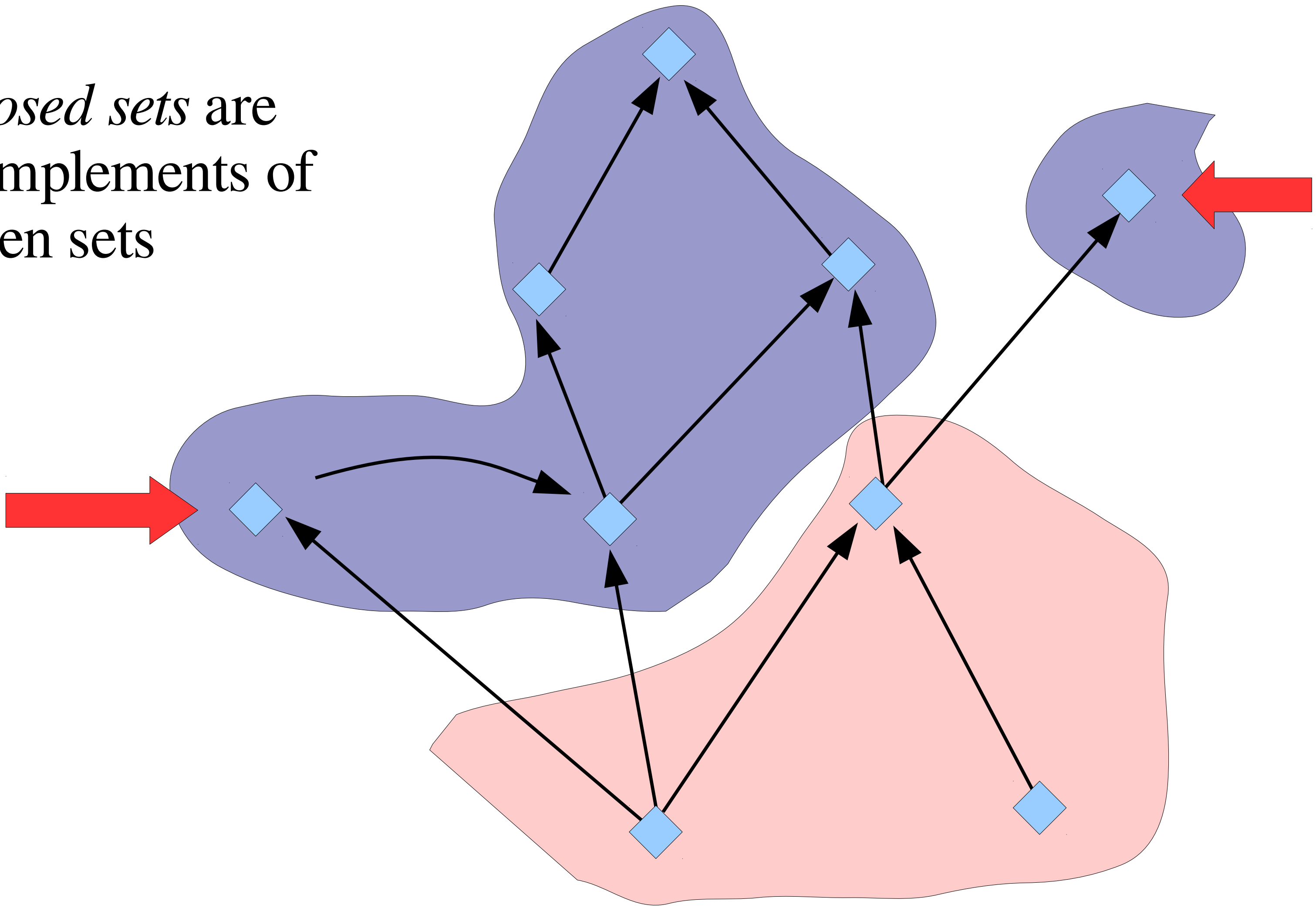
# Topologizing a partial order

Open sets are unions of *up-sets*



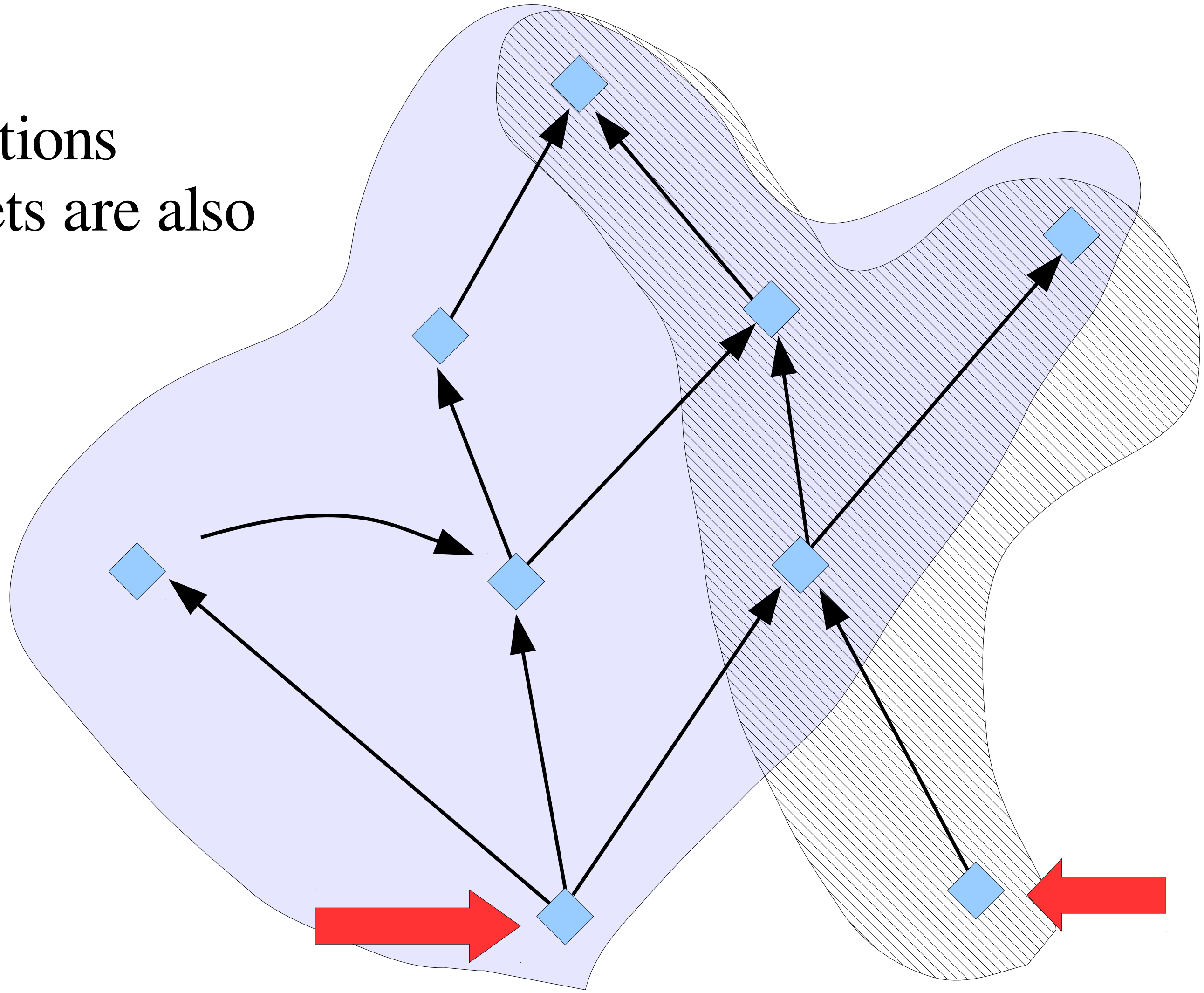
# Topologizing a partial order

*Closed sets* are complements of open sets



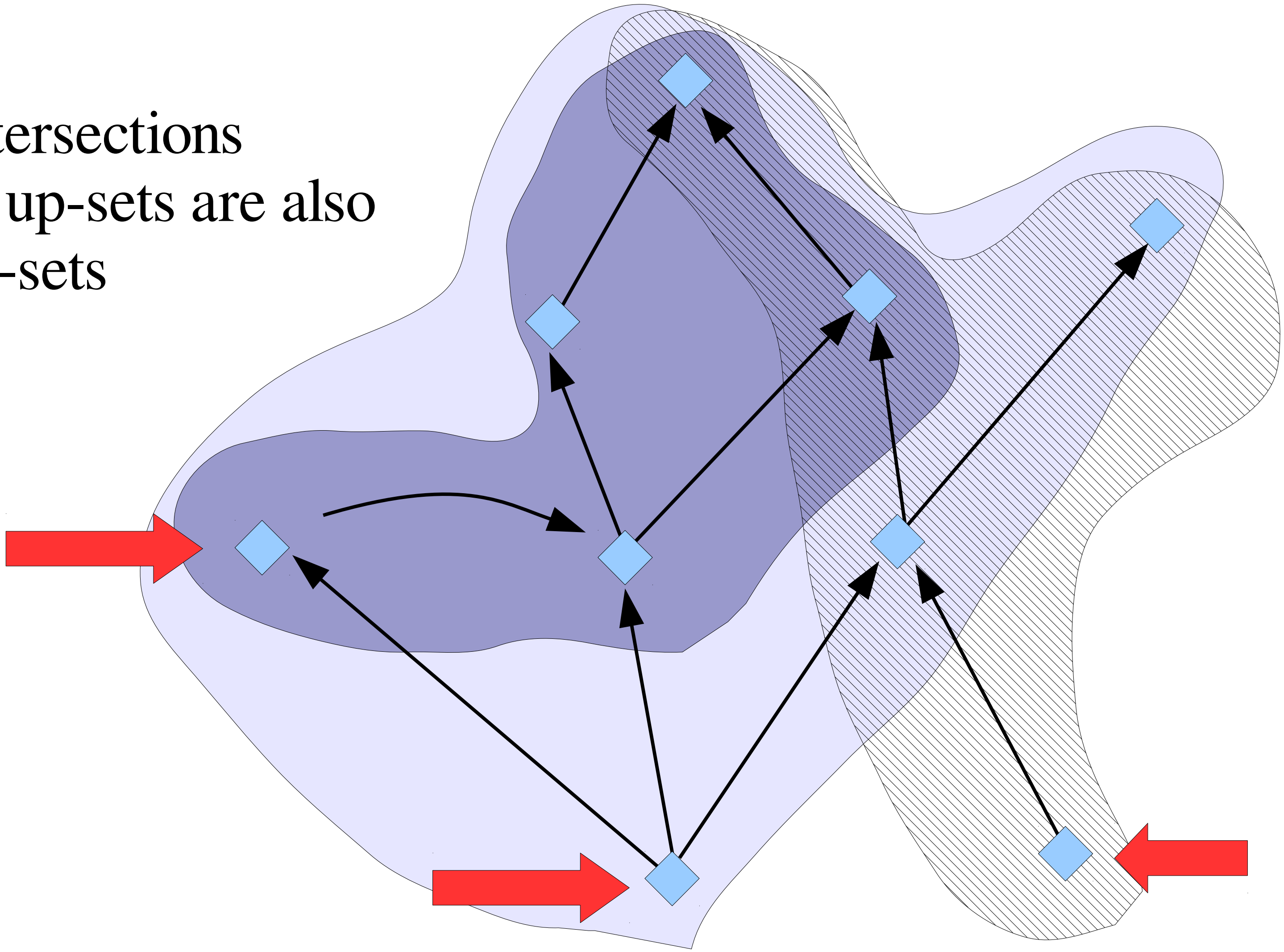
# Topologizing a partial order

Intersections of up-sets are also up-sets



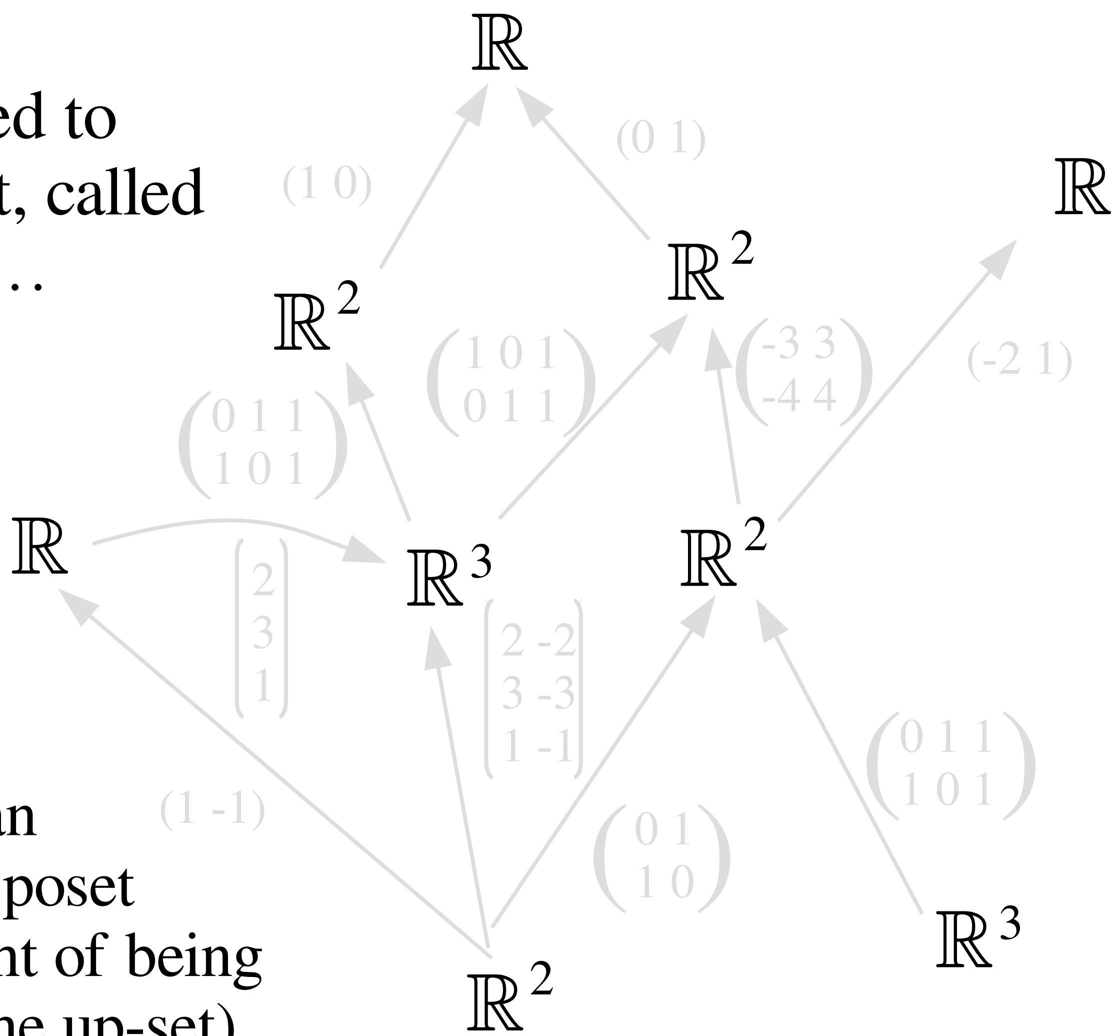
# Topologizing a partial order

Intersections of up-sets are also up-sets



# A *sheaf* on a poset is...

A set assigned to each element, called a *stalk*, and …

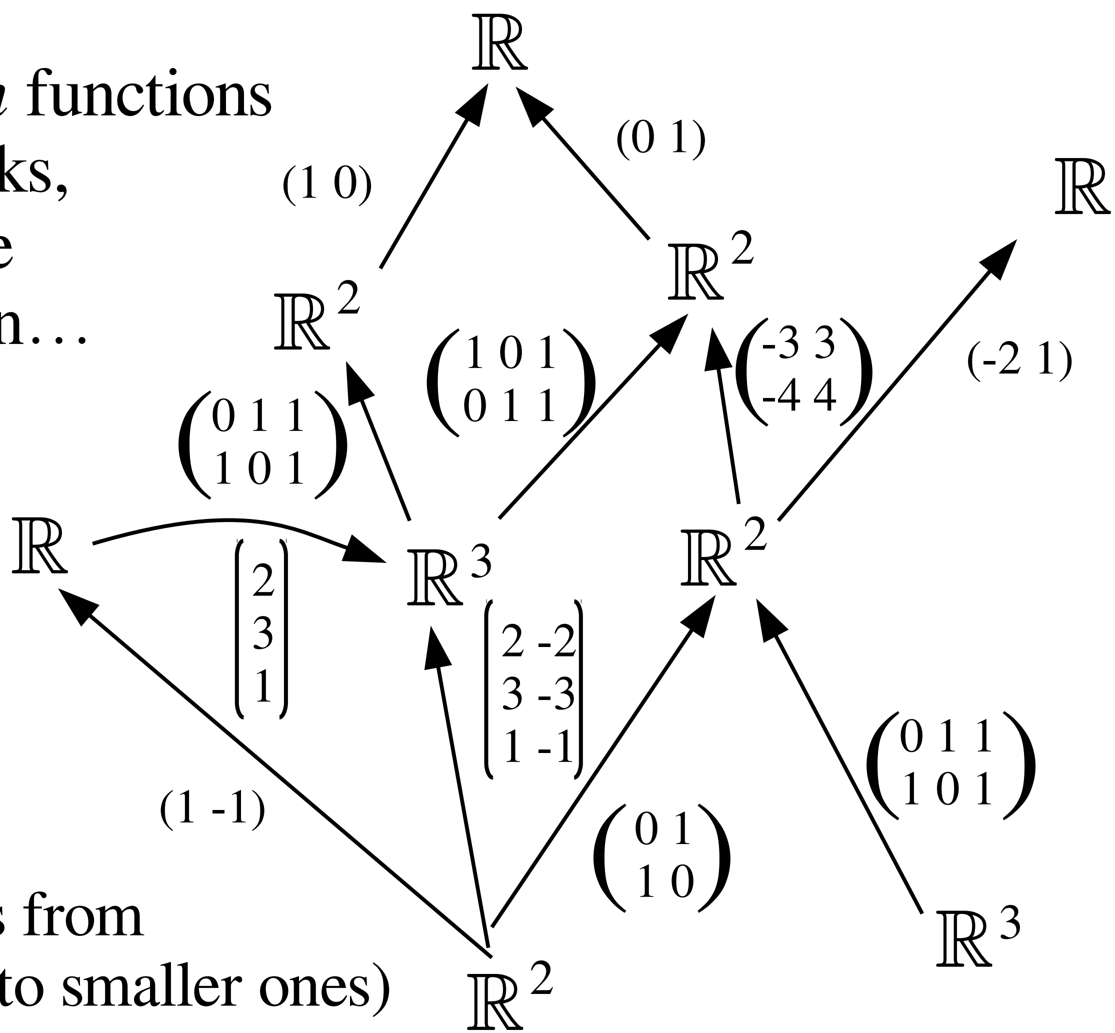



(The stalk on an element in the poset is better thought of being associated to the up-set)

This is a *sheaf* **of** vector spaces **on** a partial order

# A *sheaf* on a poset is...

… *restriction* functions between stalks, following the order relation…

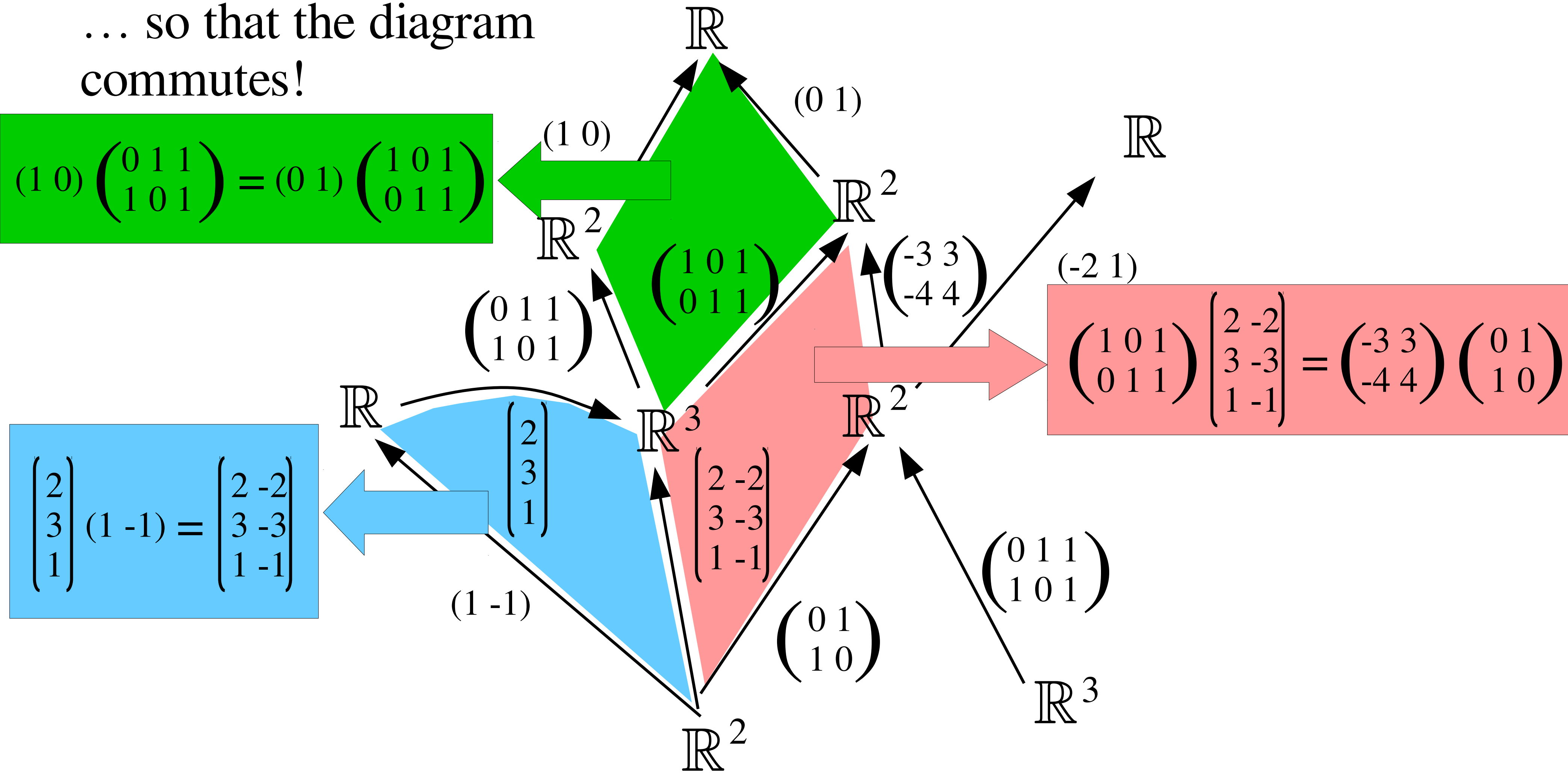



("Restriction" because it goes from bigger up-sets to smaller ones)

This is a *sheaf* **of** vector spaces **on** a partial order

# A *sheaf* on a poset is...

… so that the diagram commutes!

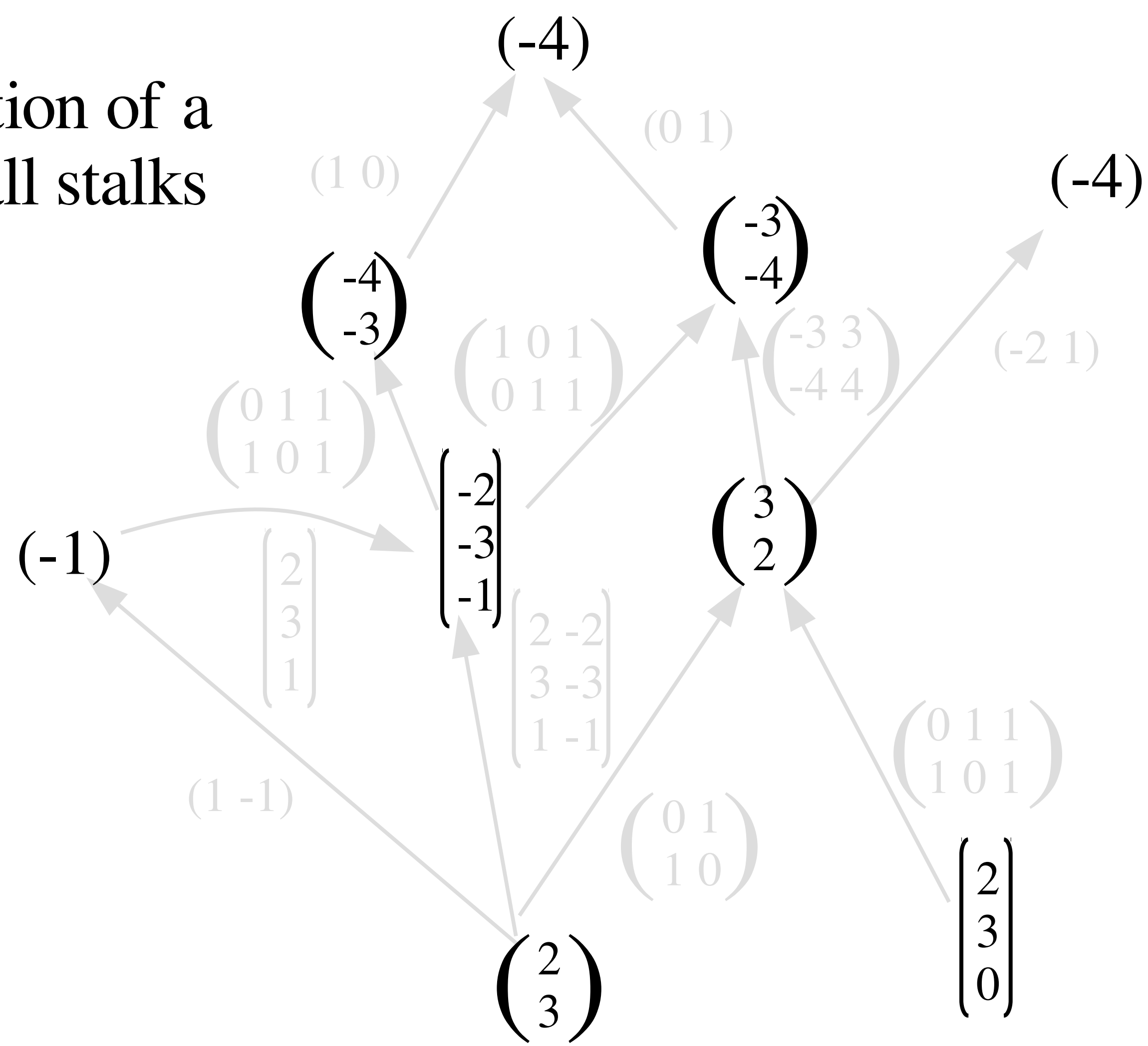



This is a *sheaf* **of** vector spaces **on** a partial order

Michael Robinson

# An *assignment* is...

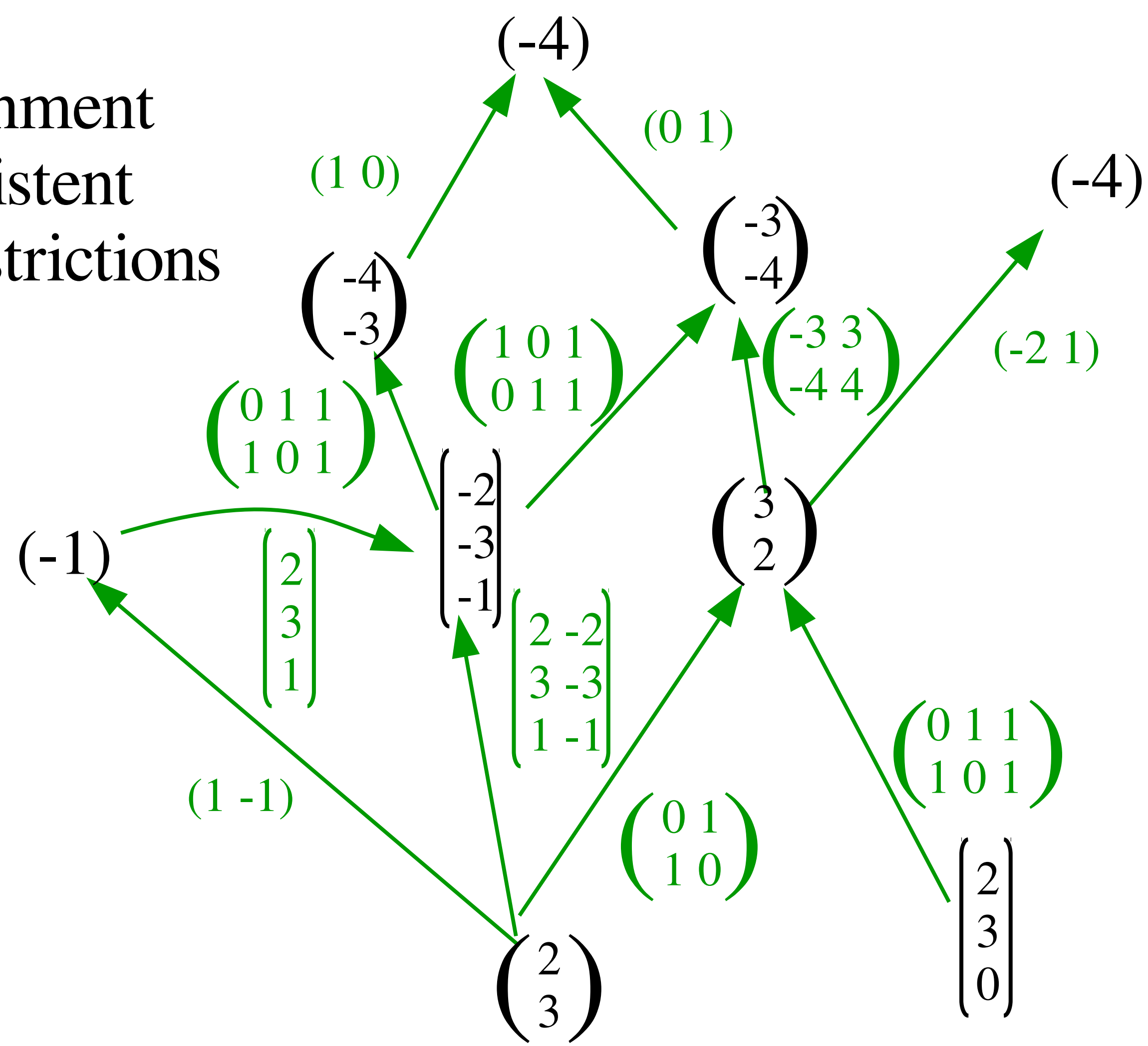
… the selection of a value from all stalks




The term *serration* is more common, but perhaps more opaque.

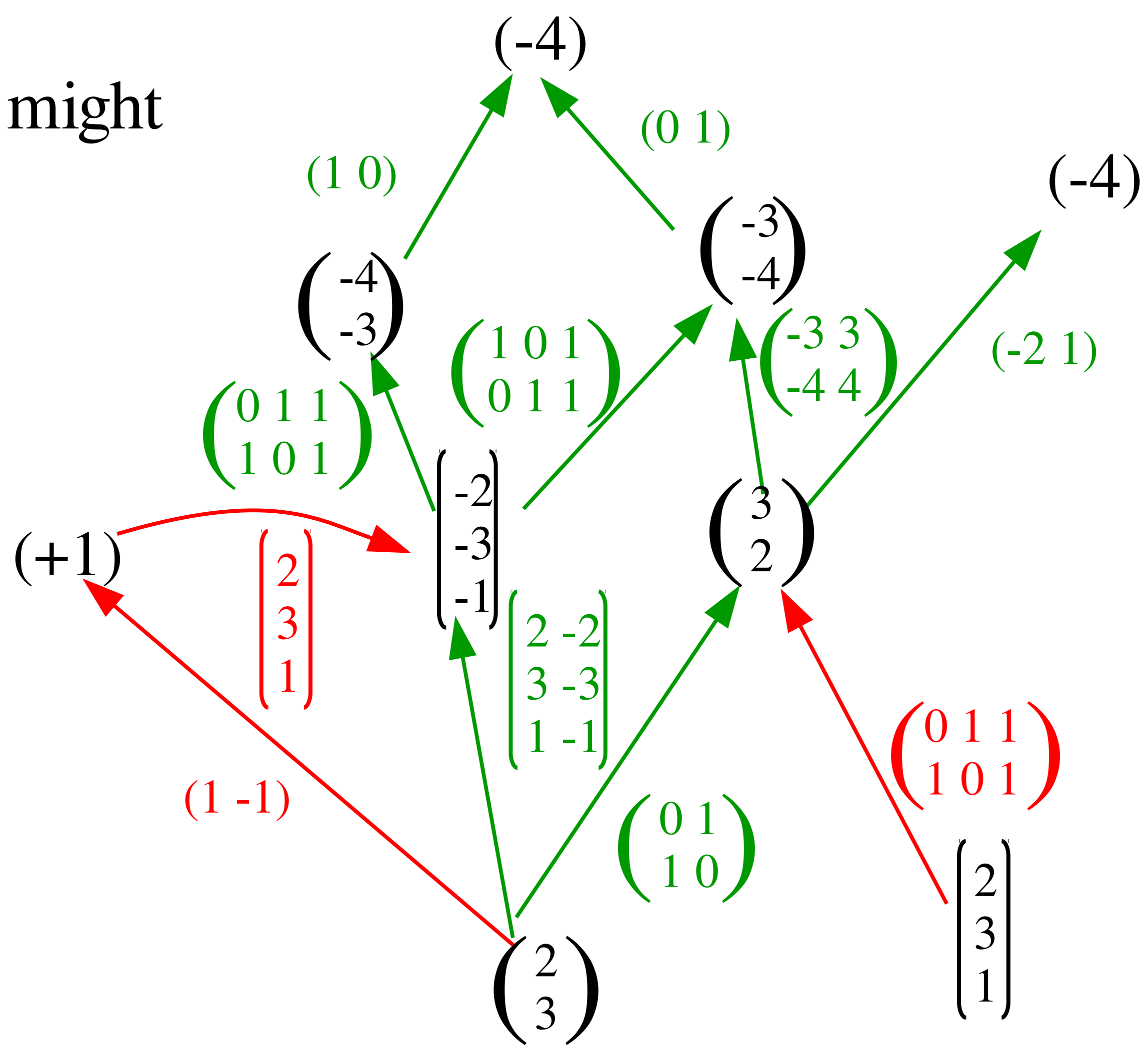
# A *global section* is...

… an assignment that is consistent with the restrictions



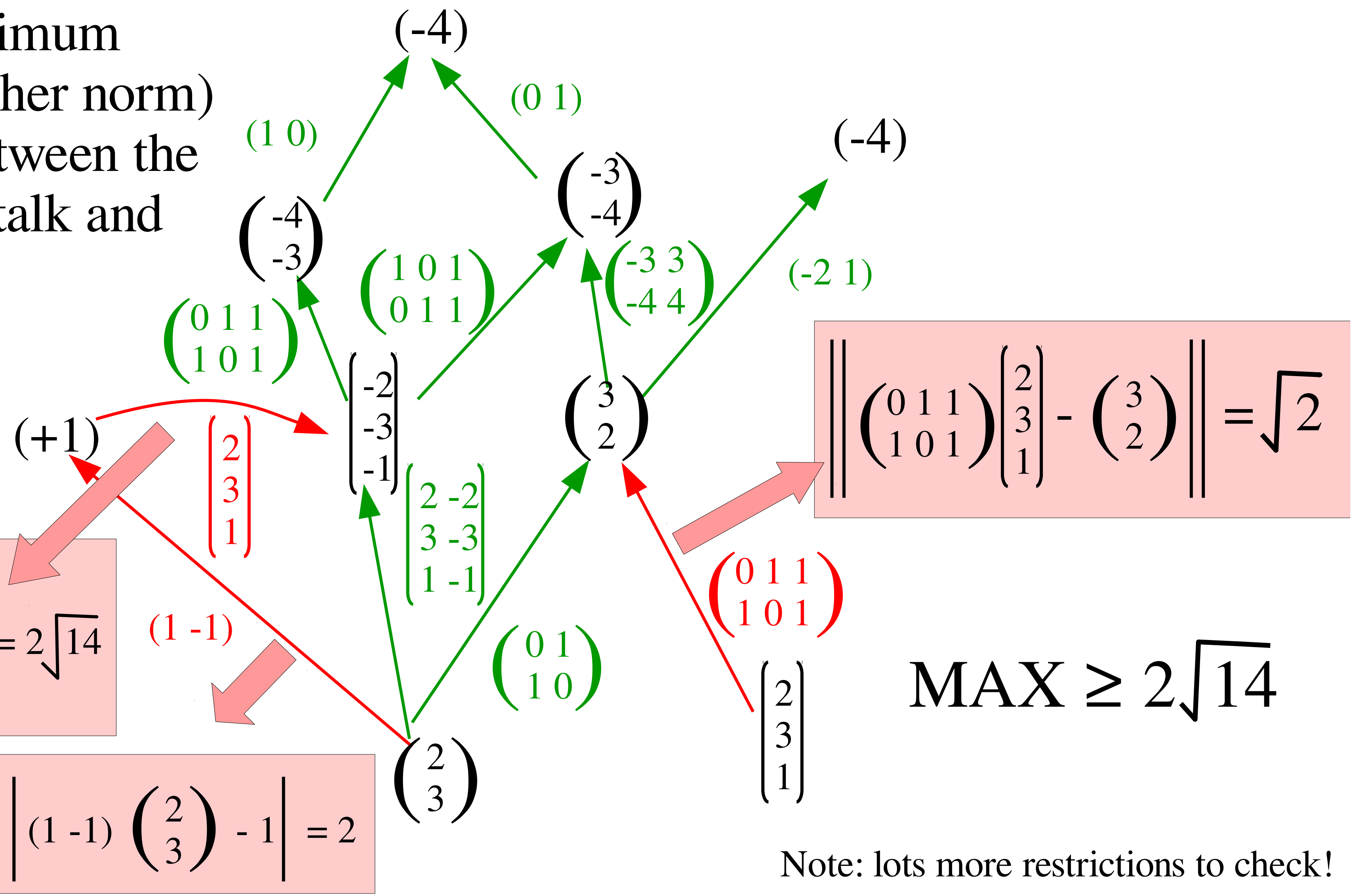
# Some assignments aren't consistent

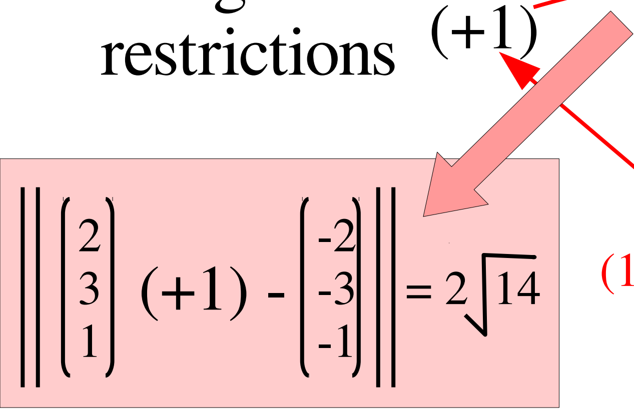
… but they might be partially consistent



# *Consistency radius* is…

… the maximum (or some other norm) distance between the value in a stalk and the values propagated along the restrictions

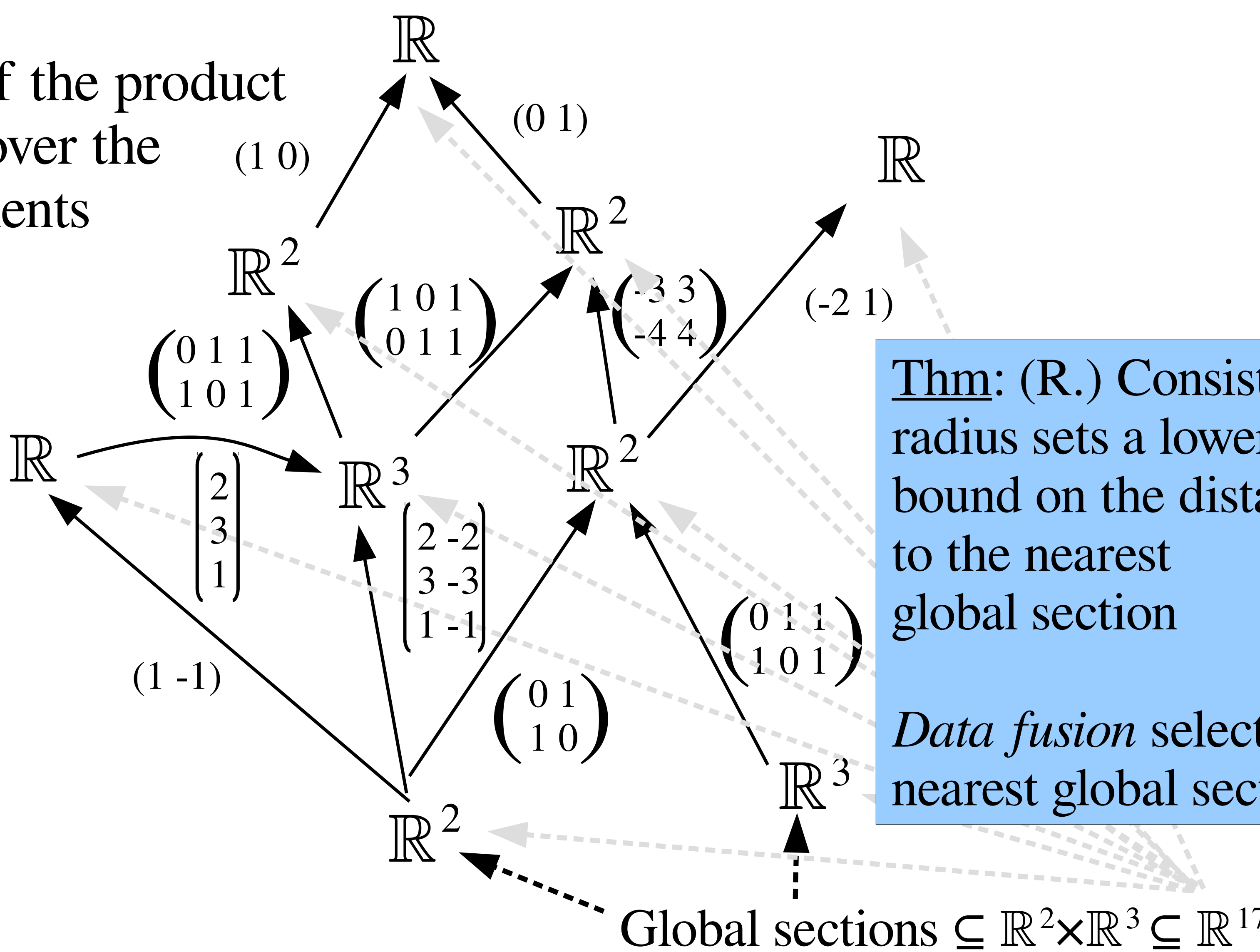



Note: lots more restrictions to check!

# The space of global sections

It's a subset of the product of the stalks over the minimal elements




Thm: (R.) Consistency radius sets a lower bound on the distance to the nearest global section

*Data fusion* selects the nearest global section

# Separating sinusoids from noise

- Consider a signal formed from *N* sinusoids
  - Each sinusoid has a (real) frequency $\omega$
  - Each sinusoid has a (complex) amplitude $a$
- Task: Recover these parameters from *M* samples

*f* is an arbitrary **known** function
Possibilities:
- Magnitude
- Phase
- Identity function
- Quantizer output
- Signal dispersion

$$x_m = f\left(\sum_{k=1}^{N} a_k e^{i\omega_k t_m} + n_m\right)$$

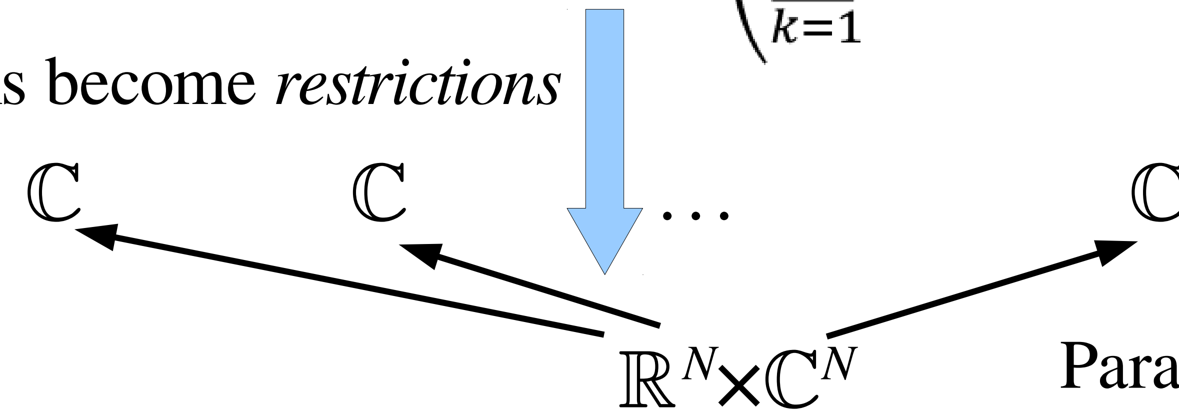
Sinusoid amplitude ($a_k$)
Sinusoid frequency ($\omega_k$)
Sample time ($t_m$)
Gaussian noise ($n_m$)

# Separating sinusoids from noise

- Consider a signal formed from $N$ sinusoids
  - Each sinusoid has a (real) frequency $\omega$
  - Each sinusoid has a (complex) amplitude $a$
- Model the situation as a *sheaf* over a poset...

$$x_m = f\left(\sum_{k=1}^{N} a_k e^{i\omega_k t_m} + n_m\right)$$

Signal models become *restrictions*



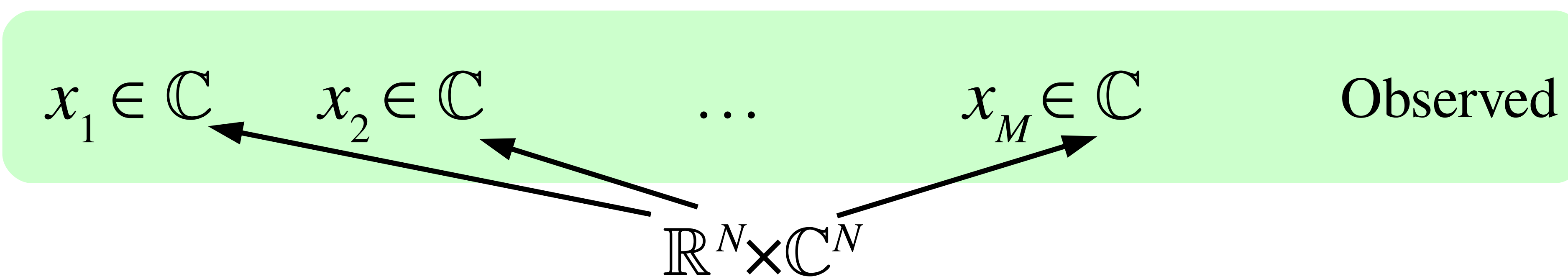

Parameter spaces become *stalks*

Michael Robinson
Chris Capraro, Janelle Henrich

# Separating sinusoids from noise

- Consider a signal formed from $N$ sinusoids
  - Each sinusoid has a (real) frequency $\omega$
  - Each sinusoid has a (complex) amplitude $a$
- The samples become an *assignment* to part of the sheaf

$$x_m = f\left(\sum_{k=1}^{N} a_k e^{i\omega_k t_m} + n_m\right)$$

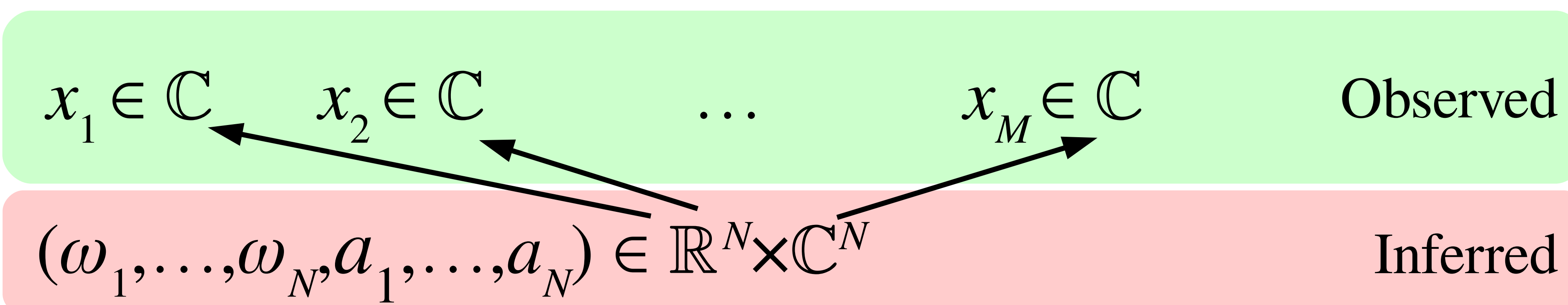



Michael Robinson
Chris Capraro, Janelle Henrich

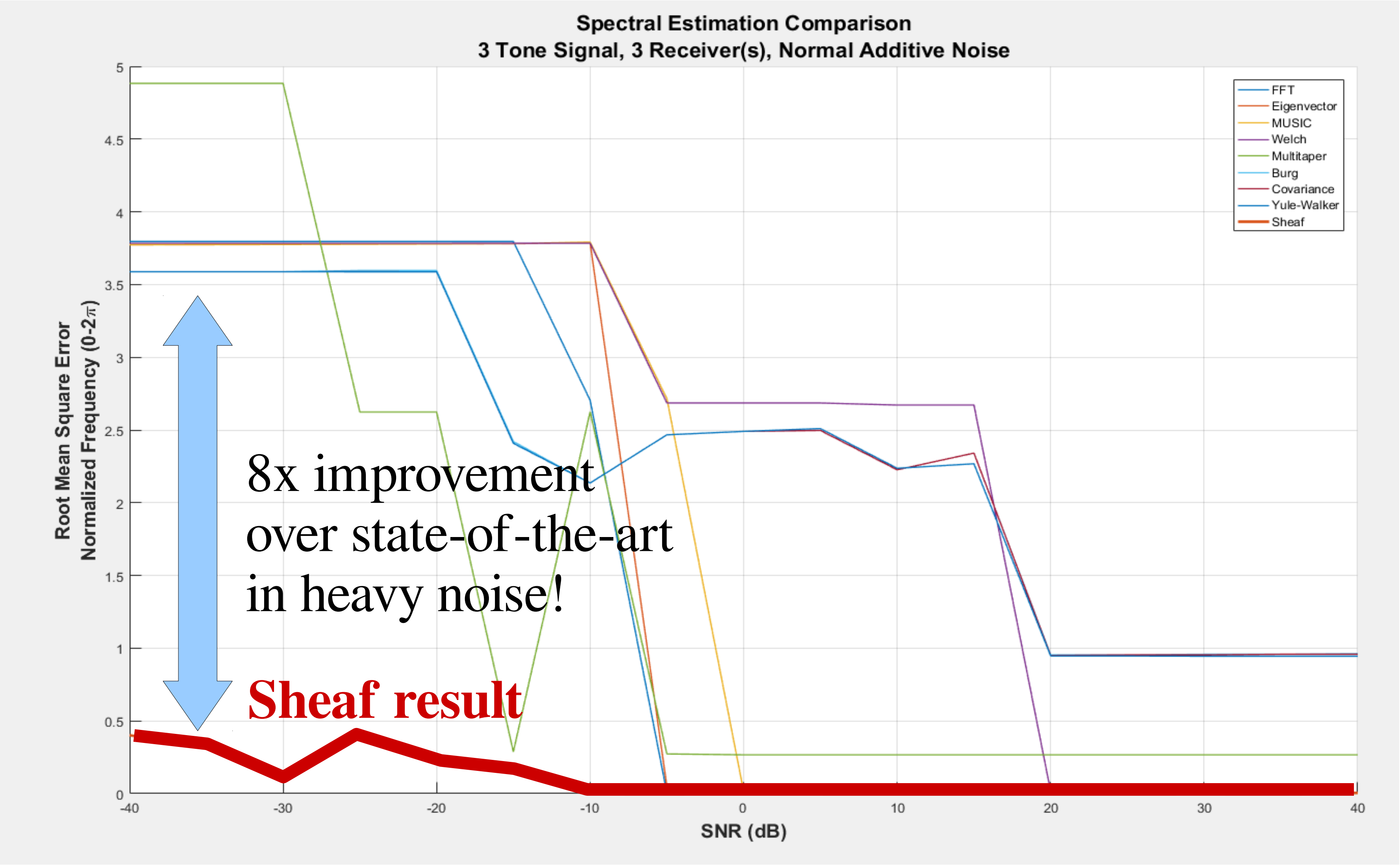
# Separating sinusoids from noise

- Consider a signal formed from $N$ sinusoids
  - Each sinusoid has a (real) frequency $\omega$
  - Each sinusoid has a (complex) amplitude $a$
- Find the unknown parameters by minimizing *consistency radius*

$$\min_{a_k, \omega_k, \text{ for } k=1,\ldots,N} \sum_{m=1}^{M} \left| f\left( \sum_{k=1}^{N} a_k e^{i\omega_k t_m} \right) - x_m \right|^2$$



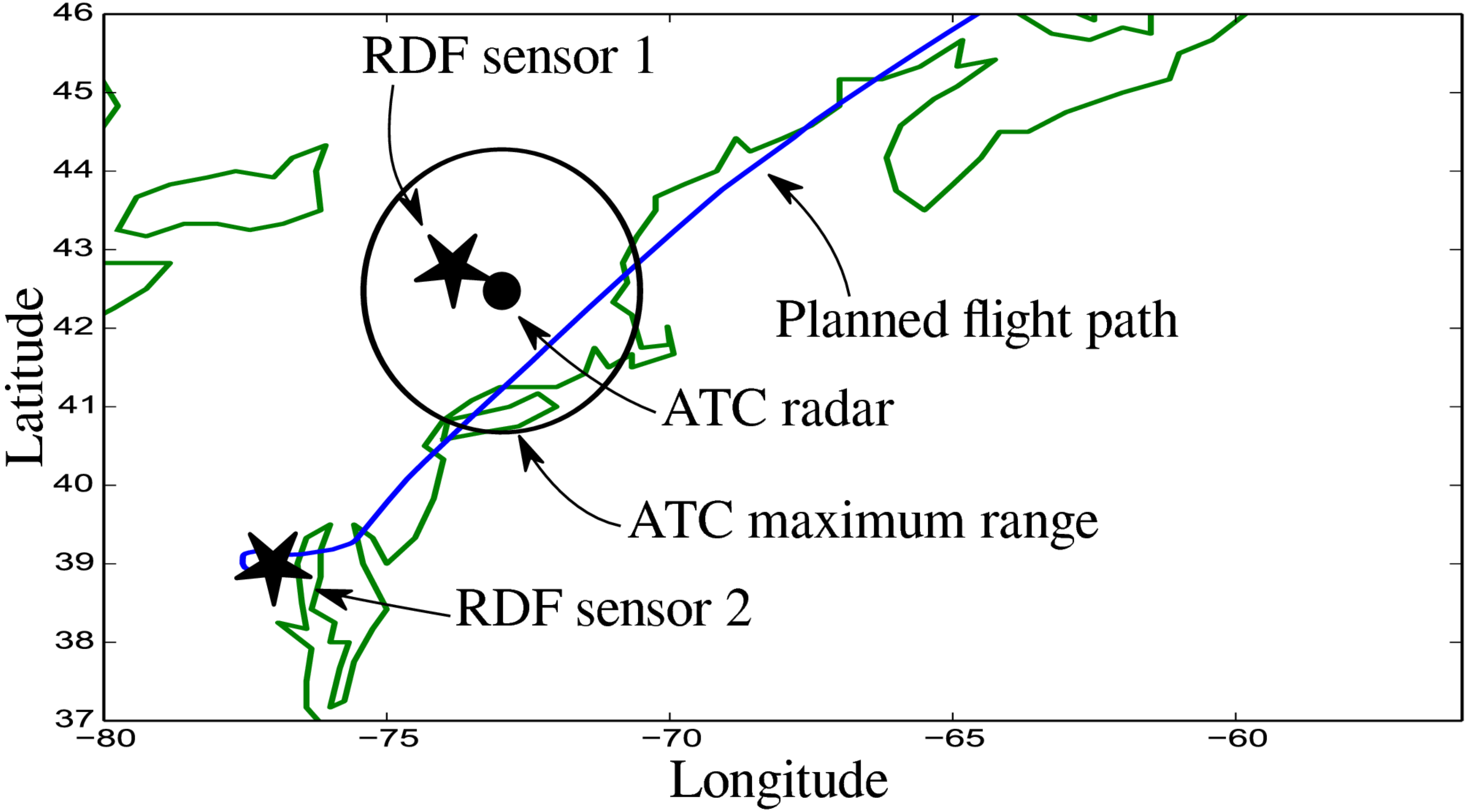
# Sheaves deliver excellent performance



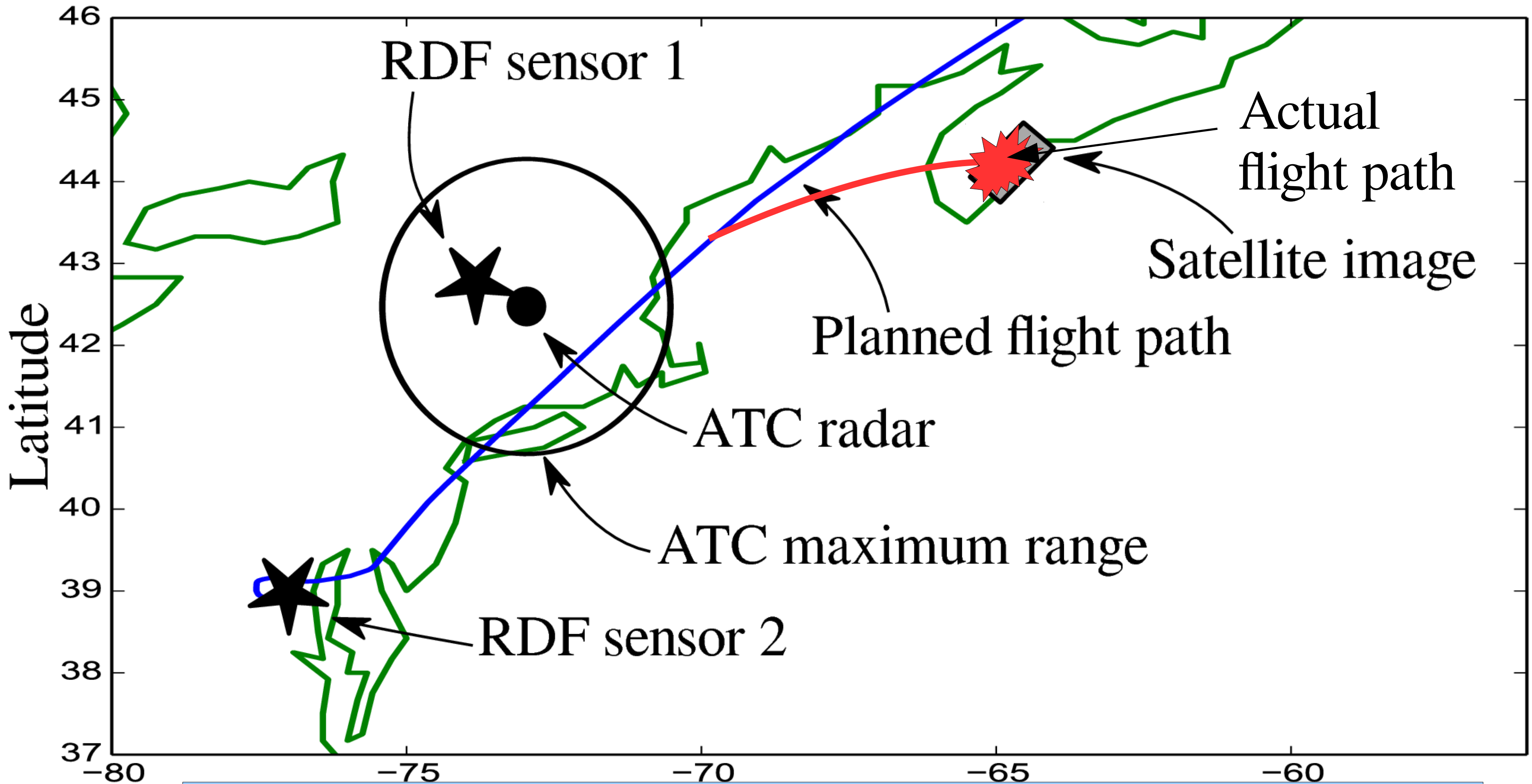


Michael Robinson
Chris Capraro, Janelle Henrich

# More complex example: flight tracking



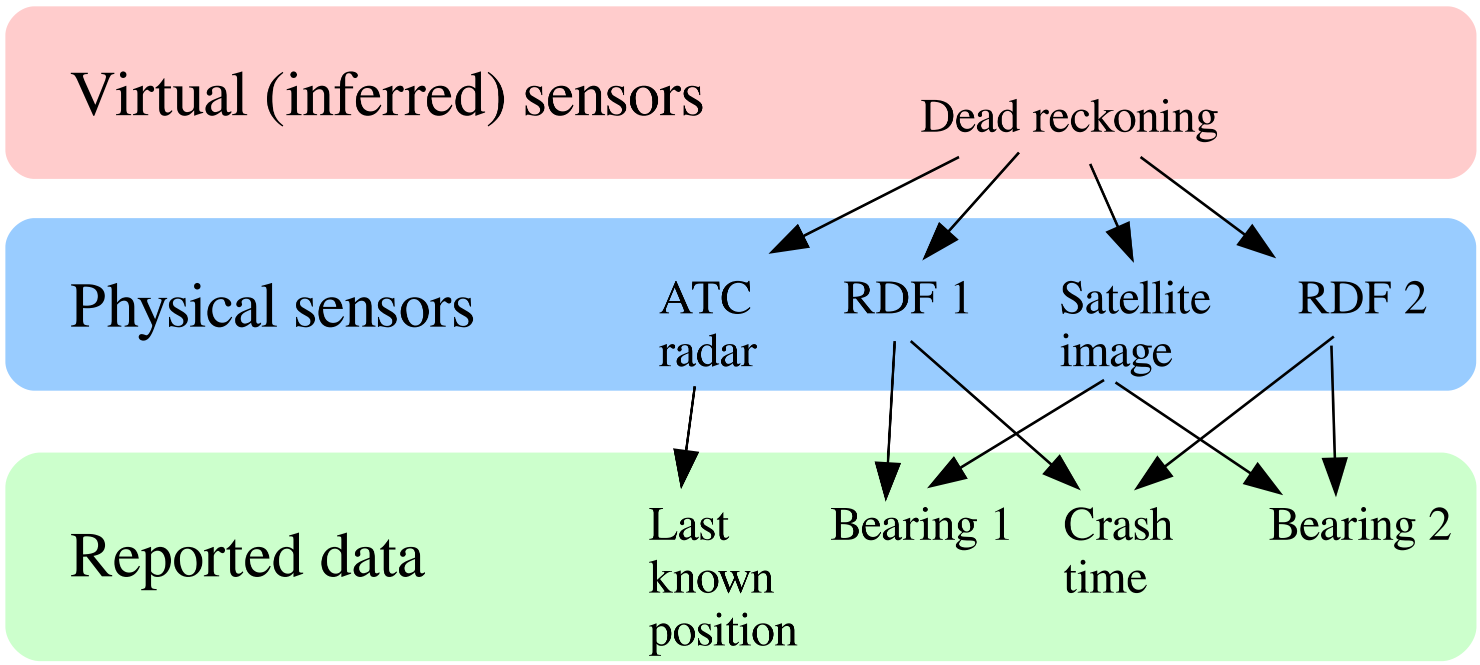
# … turns into a search and rescue mission




Observations generated using realistic simulated data…
(known crash location withheld for validation)

# Sheaf model of the sensors

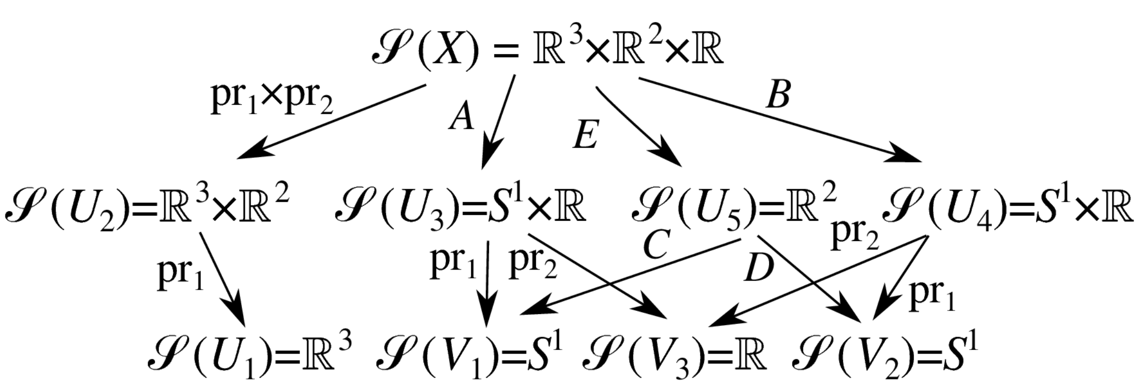
- We can form a partial order of the sensors and their overlaps



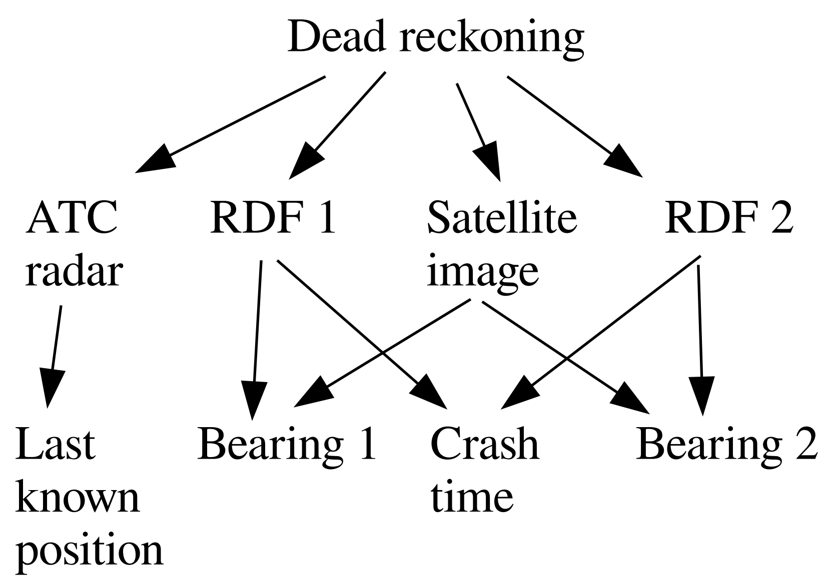

Partial order

# Sheaf model of the sensors

- We can form a partial order of the sensors and their overlaps



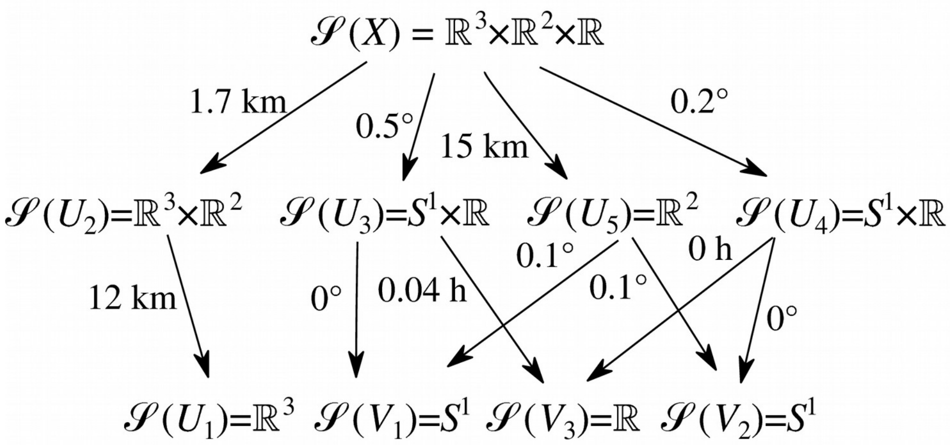

Sheaf model



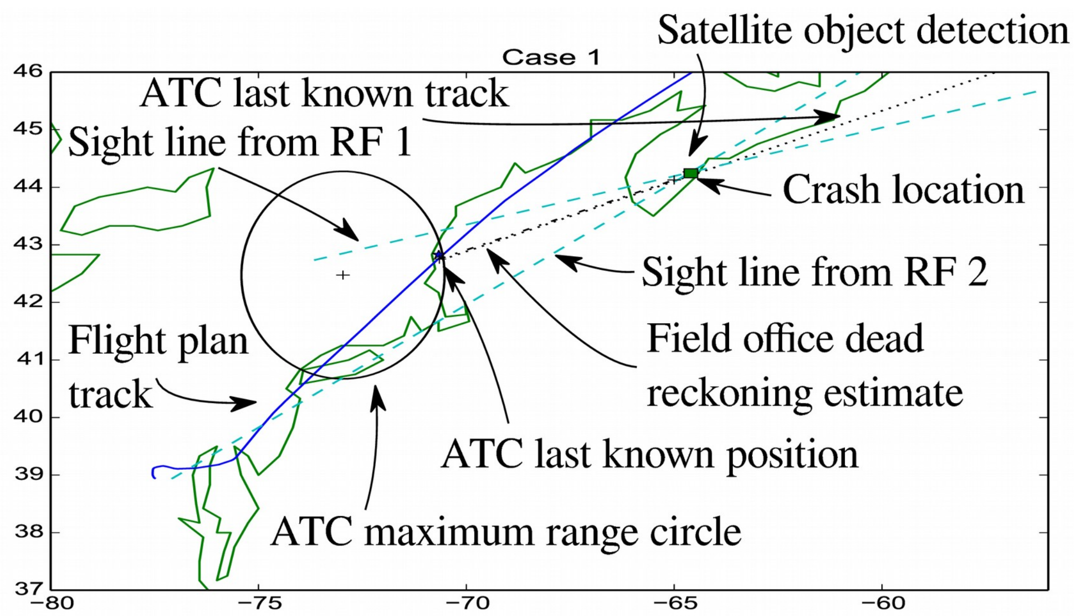

Partial order

Restrictions $A$, $B$, $C$, $D$ compute bearings from lat/lon
Restriction $E$ computes estimated crash location from last known position, velocity, time

# Case 1: Known flight path




(a)

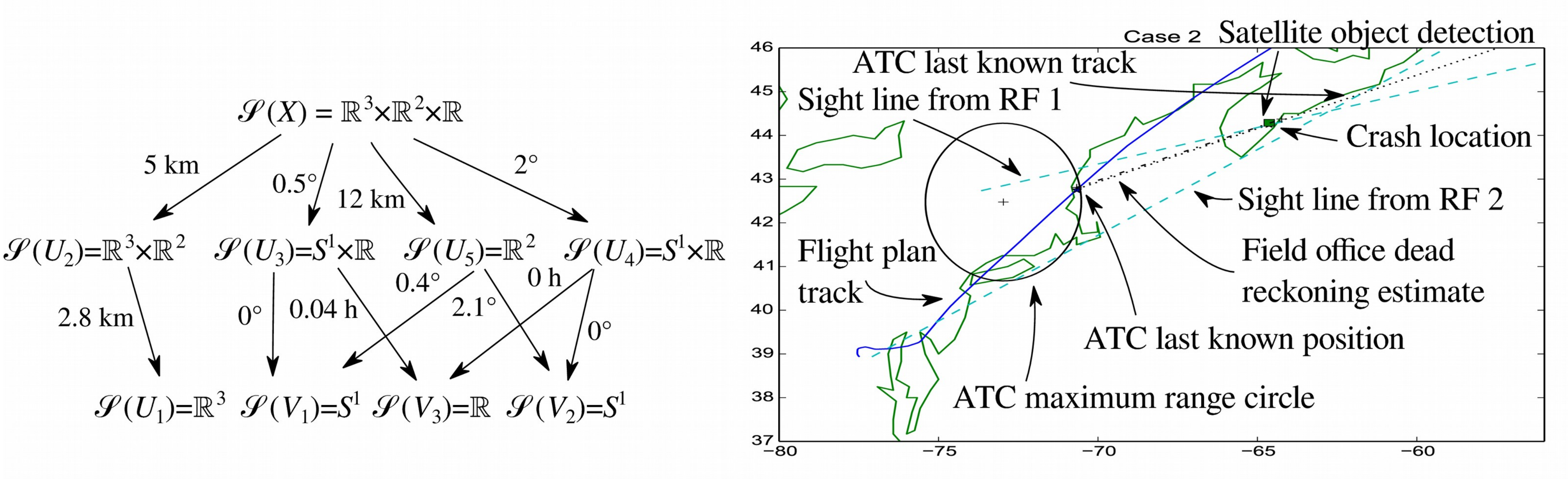



(b)

<u>Raw data:</u>
Consistency radius: 15.7 km
Crash site error:  16.1 km (using last known position only)

<u>Post-fusion:</u>
Crash site error: 2.0 km

# Case 2: Minor RDF angle error

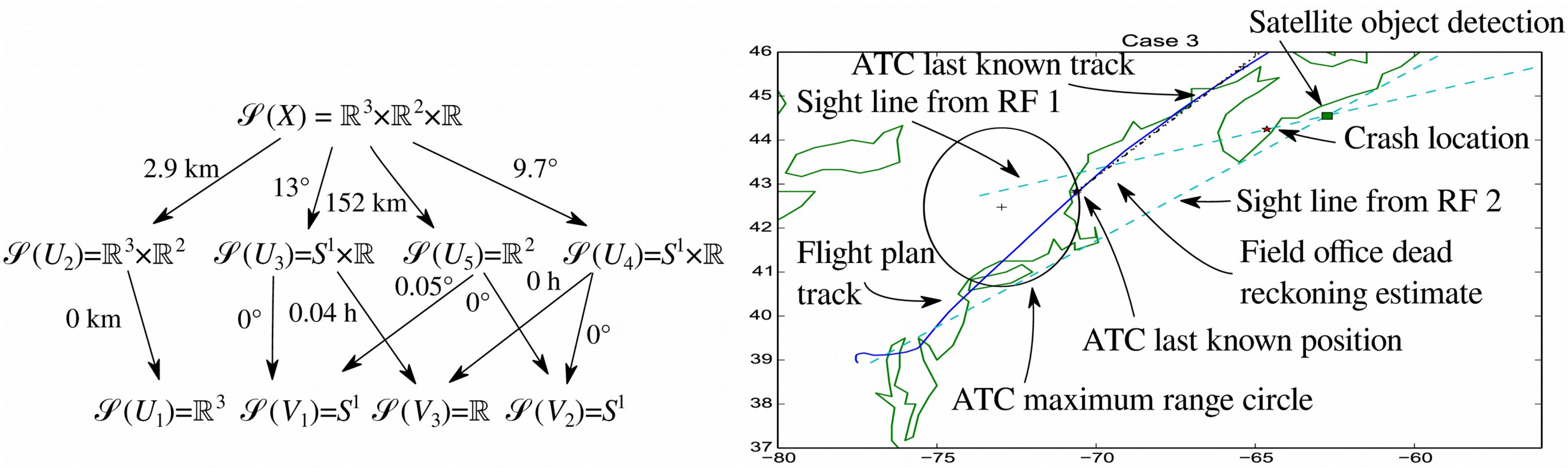



(a) (b)

<u>Raw data:</u>
Consistency radius: 11.6 km
Crash site error: 17.3 km (using last known position only)

<u>Post-fusion:</u>
Crash site error: 8.4 km

# Case 3: Major flight path error



(a) (b)

High consistency radius means data and model **are in conflict...**

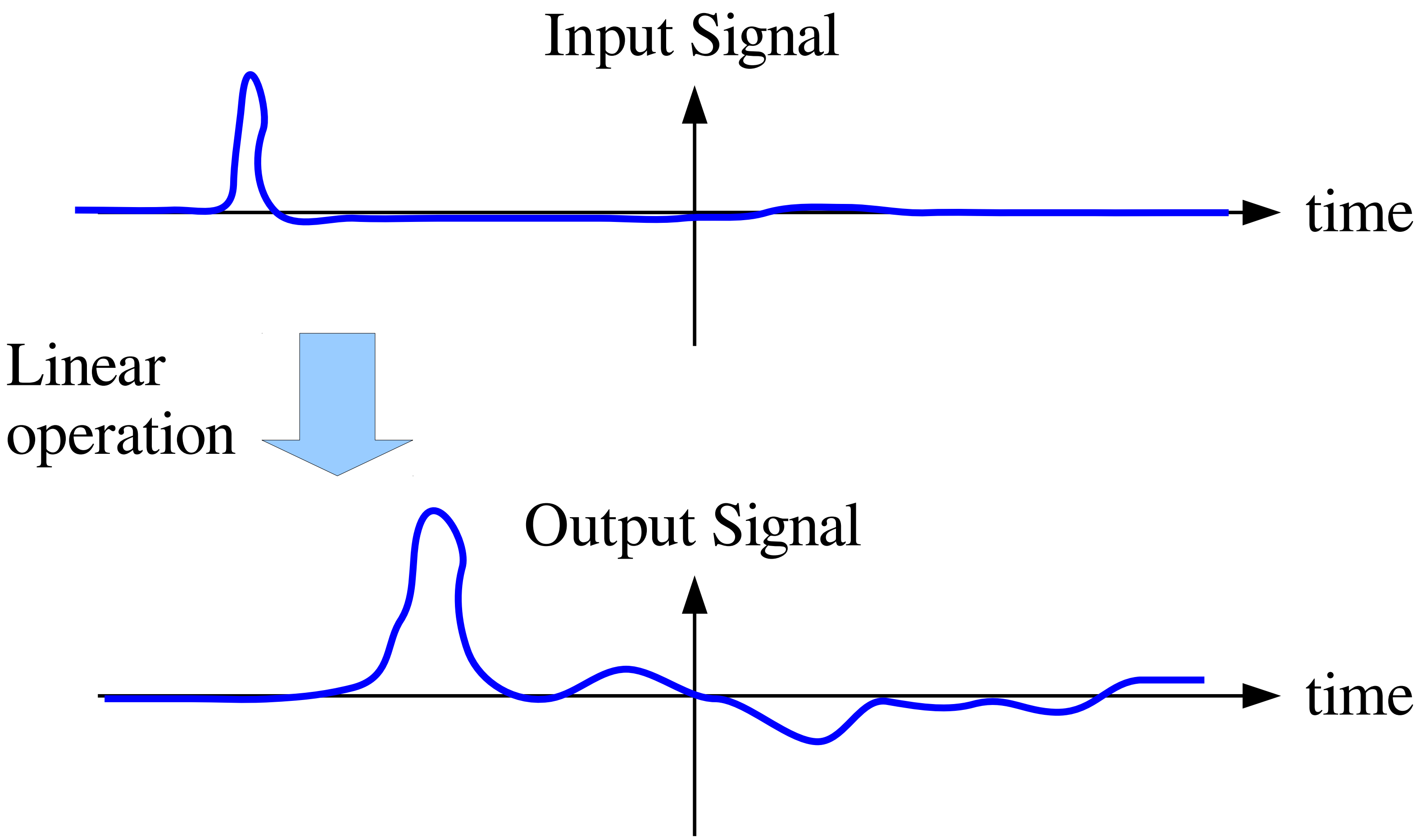
<u>Raw data:</u>
Consistency radius: 152 km
Crash site error: 193 km (using last known position only)

<u>Post-fusion:</u>
Crash site error: 74.4 km

# Topological filters

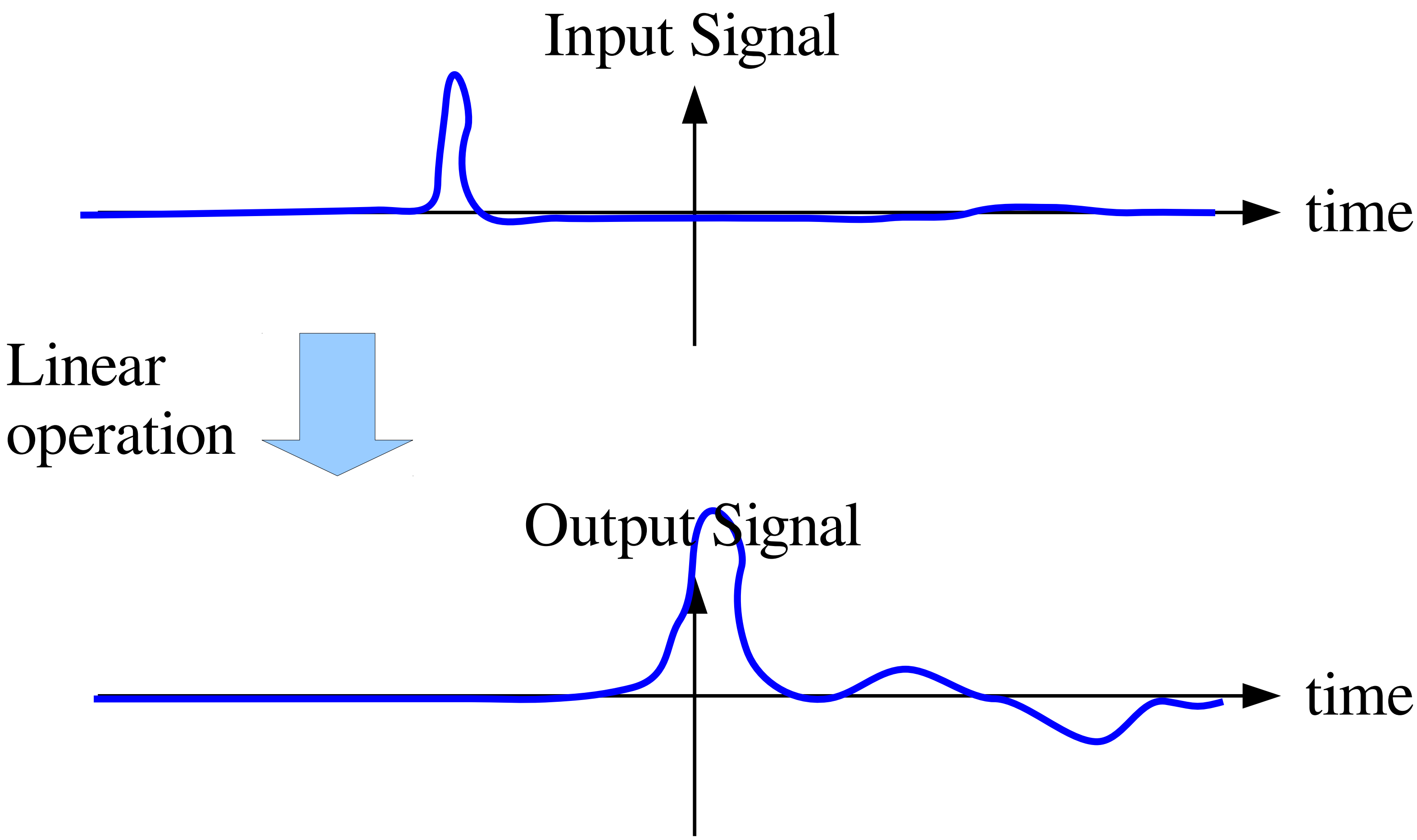
# Discrete-time LTI filters

- ***Linear*** *Translation-Invariant* filters are the workhorses of modern signal processing



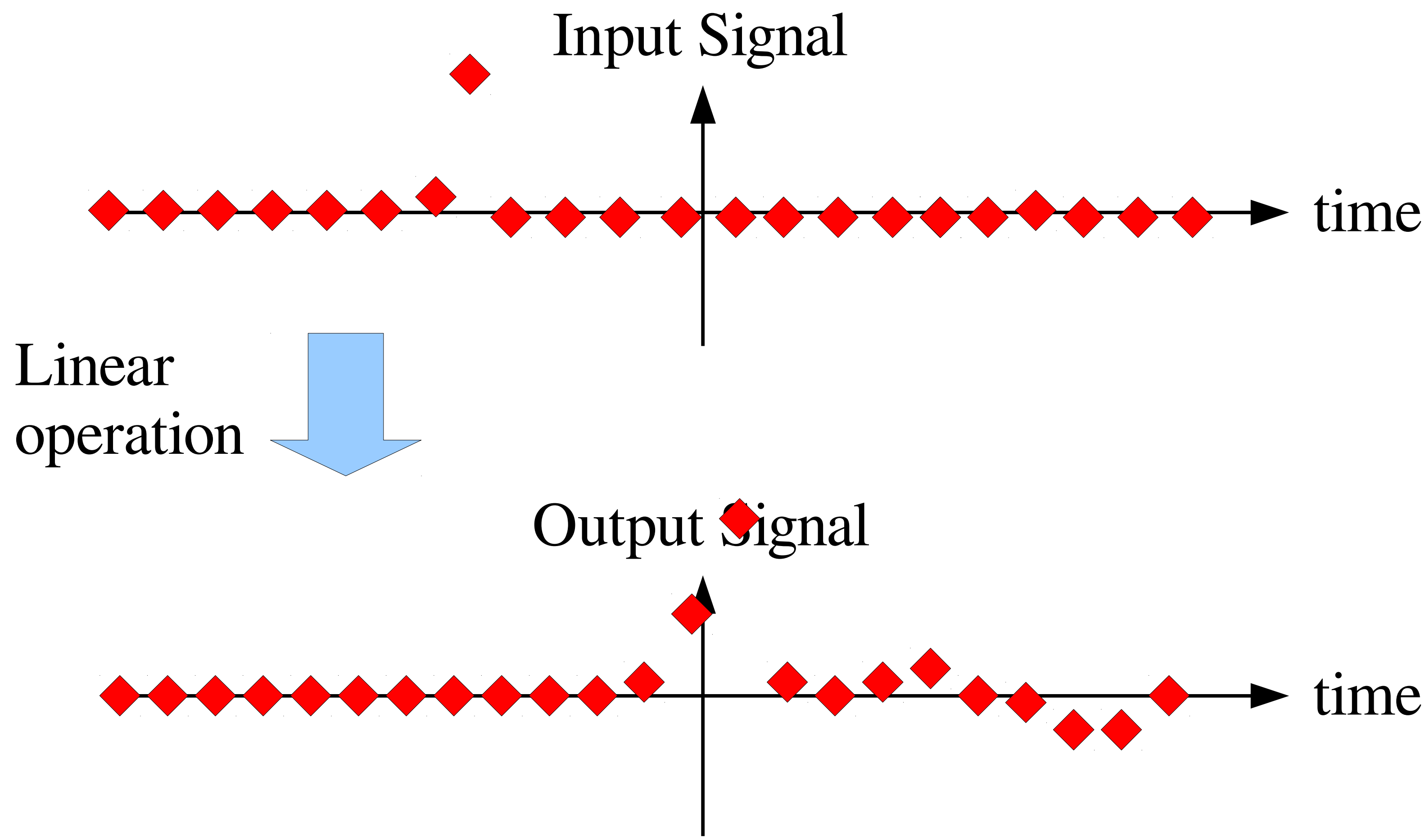
# Discrete-time LTI filters

- *Linear* ***Translation-Invariant*** filters are the workhorses of modern signal processing



# **Discrete-time** LTI filters

- *Linear Translation-Invariant* filters are the workhorses of modern signal processing



# Filters as sheaf morphisms

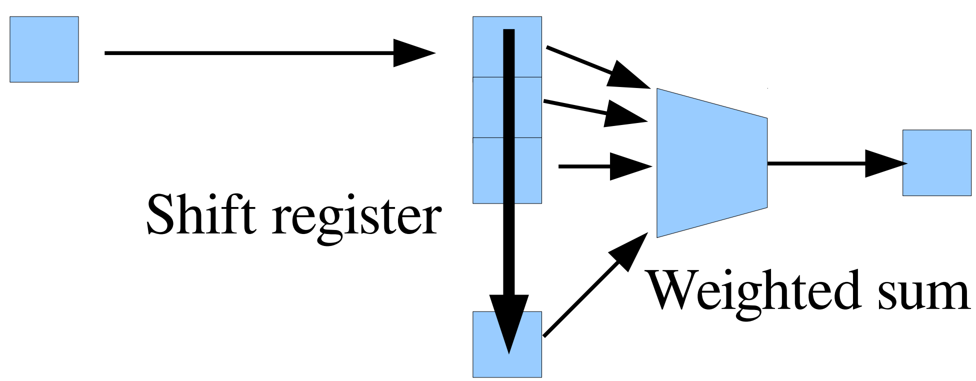
- <u>Theorem</u>: Every discrete-time LTI filter can be encoded as a sequence of two sheaf morphisms

$$S_1 \xleftarrow{\text{projection}} S_2 \xrightarrow{\text{combination}} S_3$$

Sheaf formalism

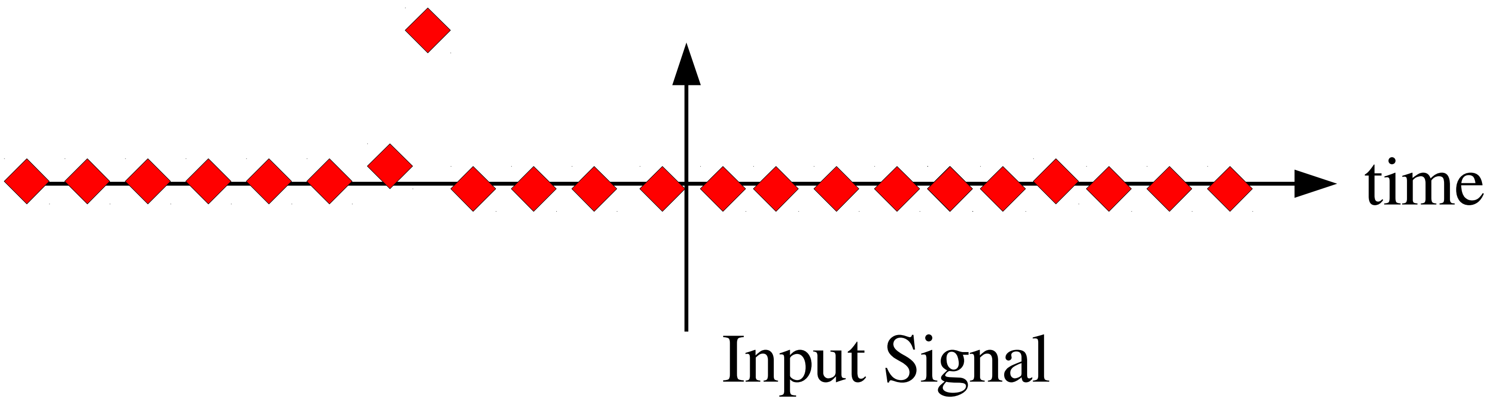
Input — Internal state — Output

Hardware



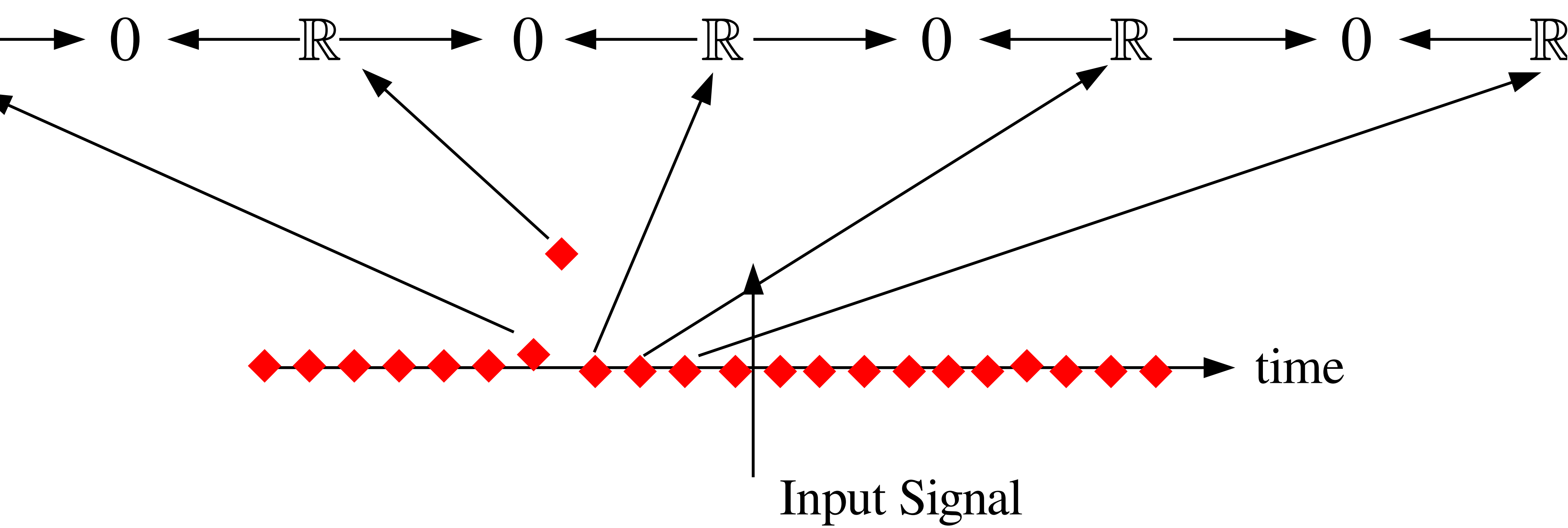
# Proof sketch: Input sheaf

- Sections of this sheaf are timeseries, instead of continuous functions



# Proof sketch: Input sheaf

- Sections of this sheaf are timeseries, instead of continuous functions



# Proof sketch: Input sheaf

- Sections of this sheaf are timeseries, instead of continuous functions

$$\rightarrow 0 \leftarrow \mathbb{R} \rightarrow 0 \leftarrow \mathbb{R} \rightarrow 0 \leftarrow \mathbb{R} \rightarrow 0 \leftarrow \mathbb{R}$$

# Proof sketch: Output sheaf

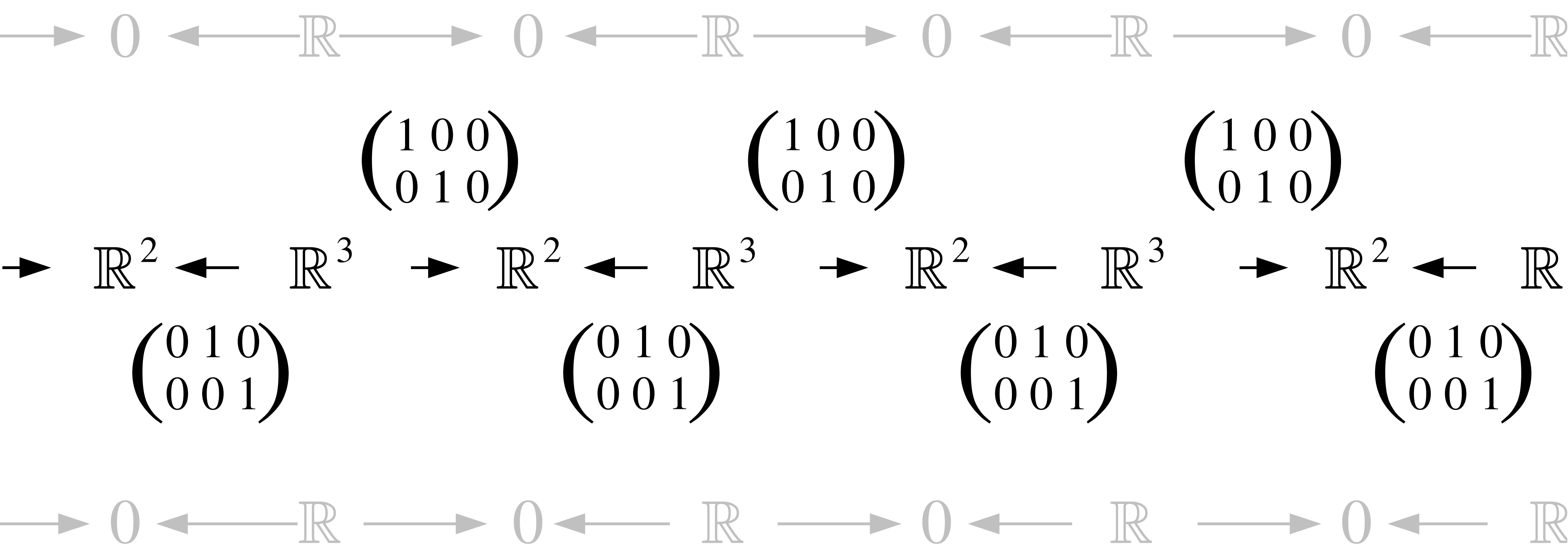
- The output sheaf is the same

$$\longrightarrow 0 \longleftarrow \mathbb{R} \longrightarrow 0 \longleftarrow \mathbb{R} \longrightarrow 0 \longleftarrow \mathbb{R} \longrightarrow 0 \longleftarrow \mathbb{R}$$

$$\longrightarrow 0 \longleftarrow \mathbb{R} \longrightarrow 0 \longleftarrow \mathbb{R} \longrightarrow 0 \longleftarrow \mathbb{R} \longrightarrow 0 \longleftarrow \mathbb{R}$$

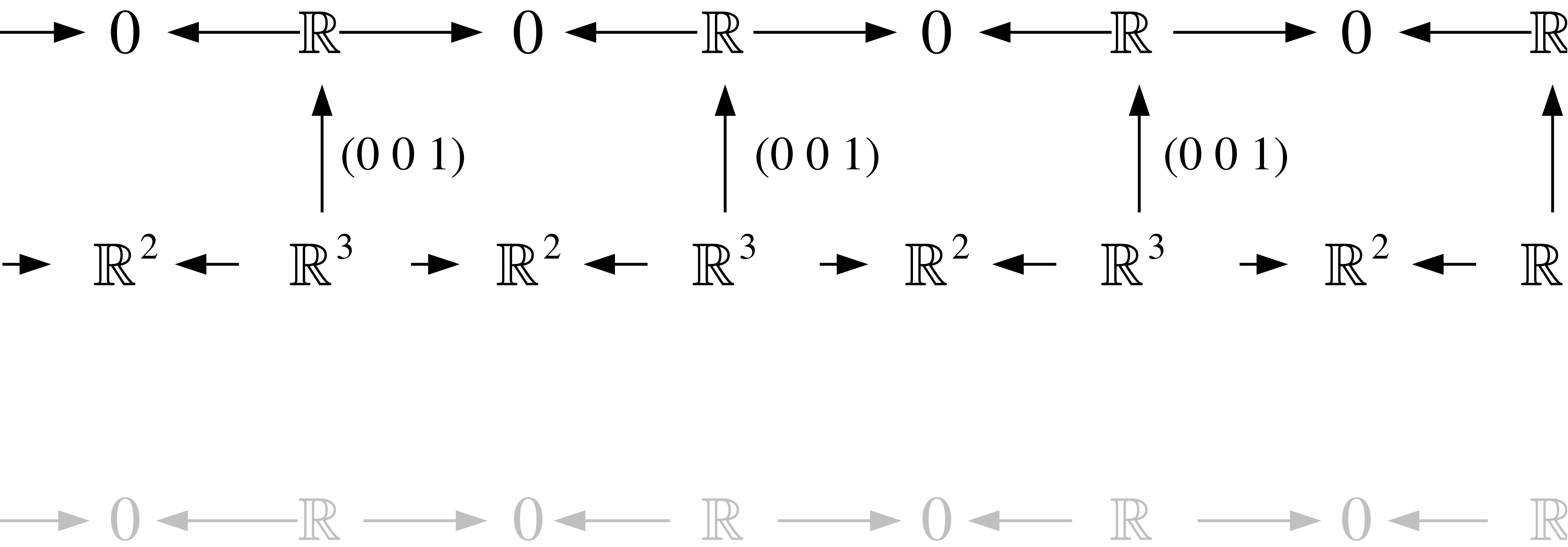
# Proof sketch: The internal state

- Contents of the shift register at each timestep
- $N = 3$ shown



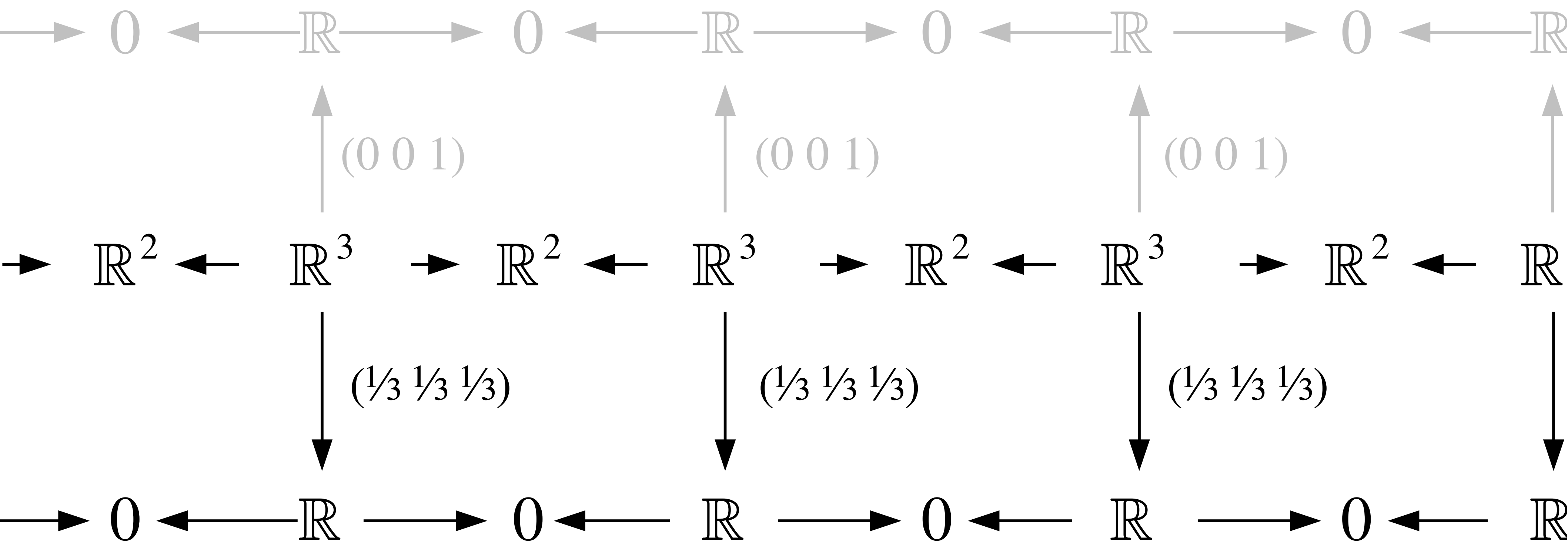
# Proof sketch: The internal state

- Loads a new value with each timestep



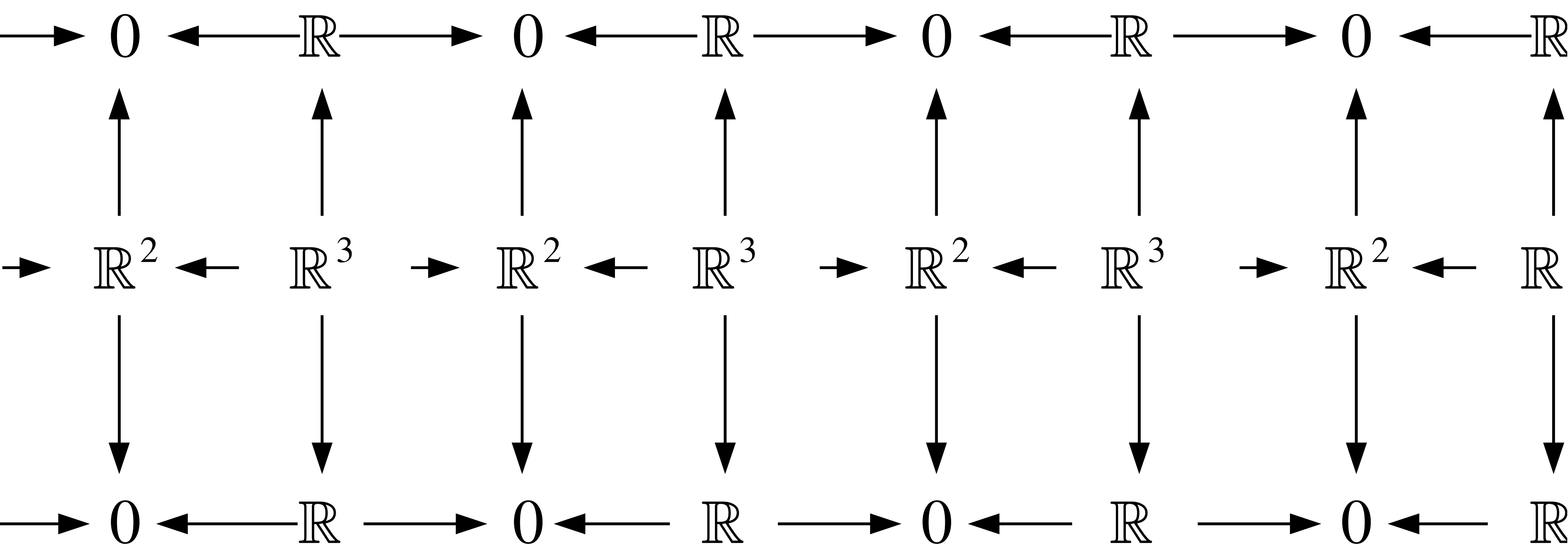
# Proof sketch: The internal state

- Computes linear functional of the shift register at each timestep (for instance, compute the mean)



# Proof sketch: Finishing both morphisms

- Put in a few zero maps!



# A practical topological filter

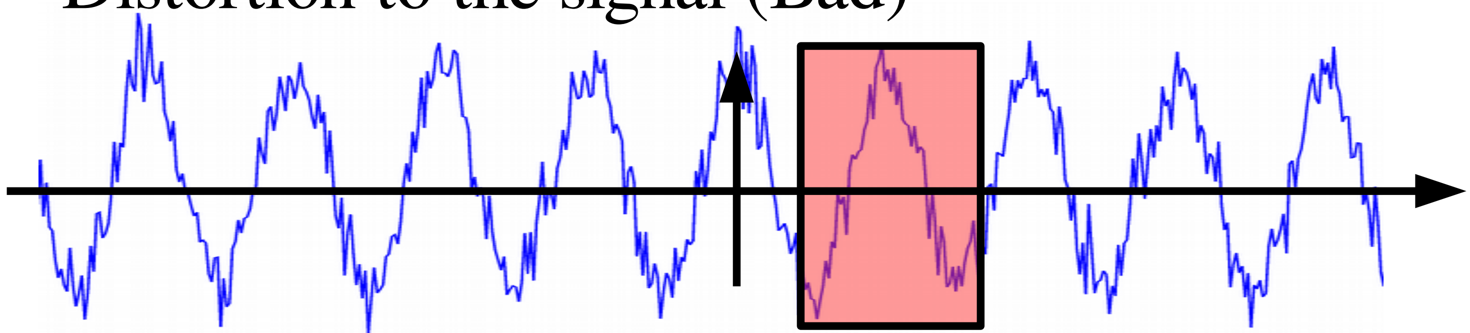
The *QuasiPeriodic Low Pass Filter*

(QPLPF)

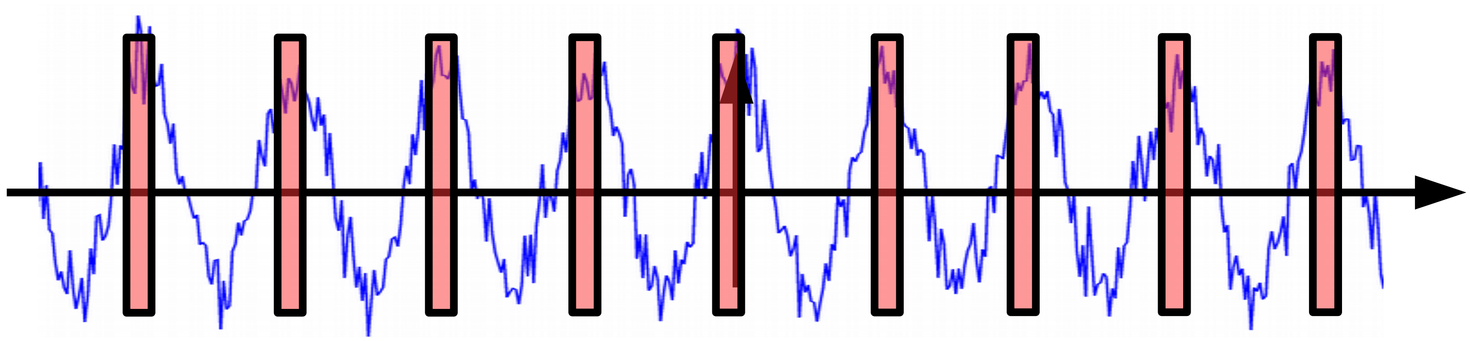
Michael Robinson

# Circumventing bandwidth limits

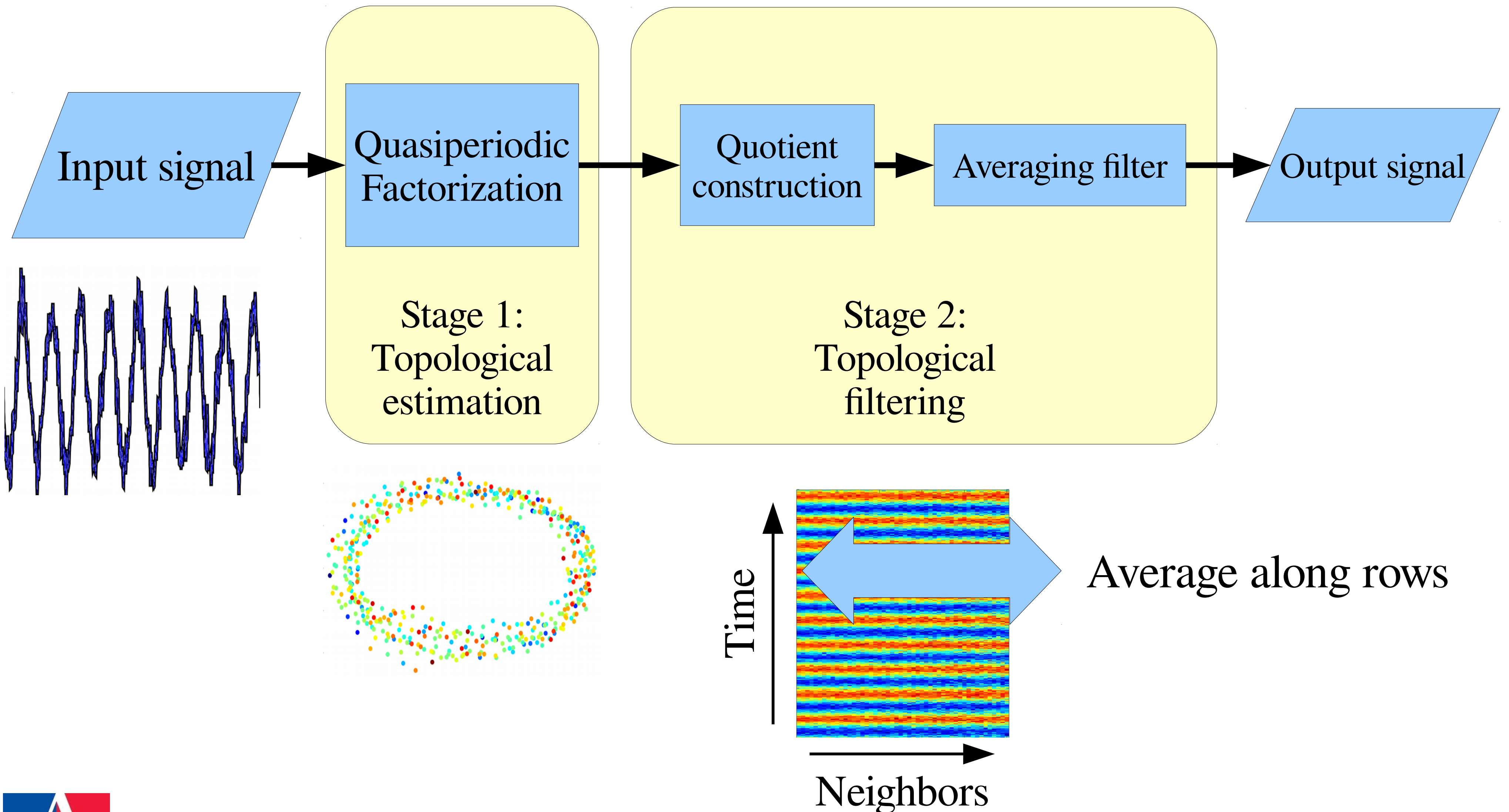
- Traditional: averaging in a connected window
  - Noise cancellation (Good)
  - Distortion to the signal (Bad)



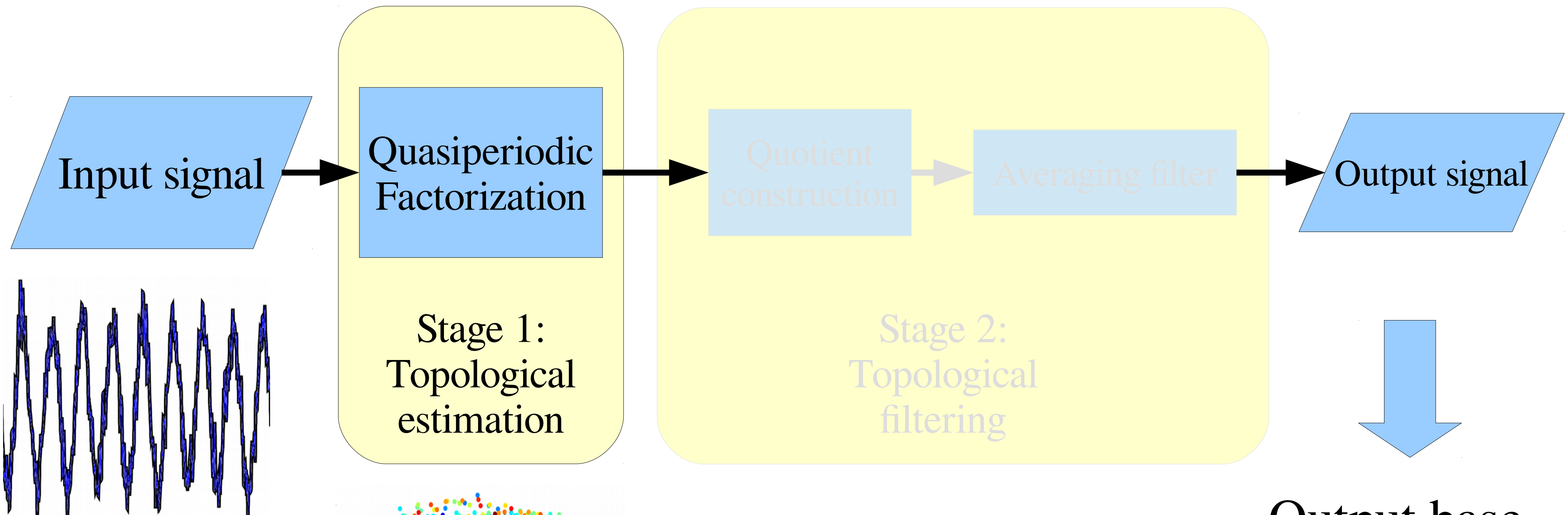
- Knowledge of the phase space: can **safely** do **more** averaging across the **entire** signal

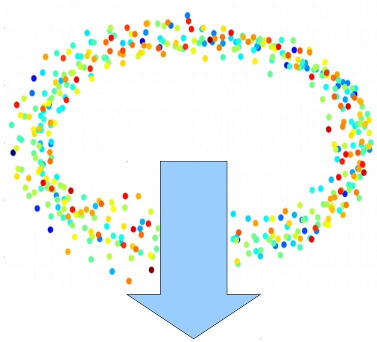


# QPLPF block diagram



# How is this a topological filter?



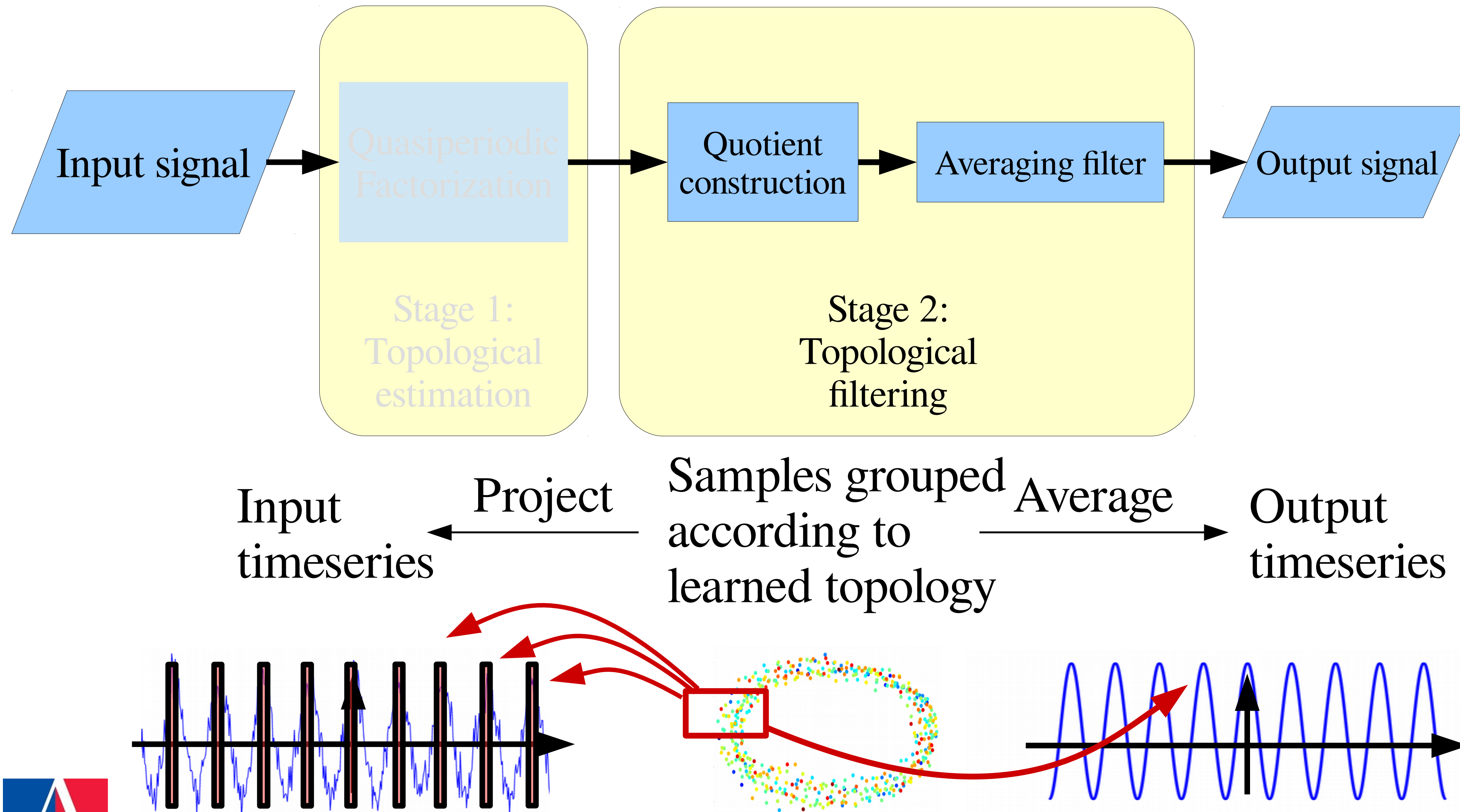



Input base space is $\mathbb{Z}$

Internal state base space is learned from the data

Output base space is $\mathbb{Z}$

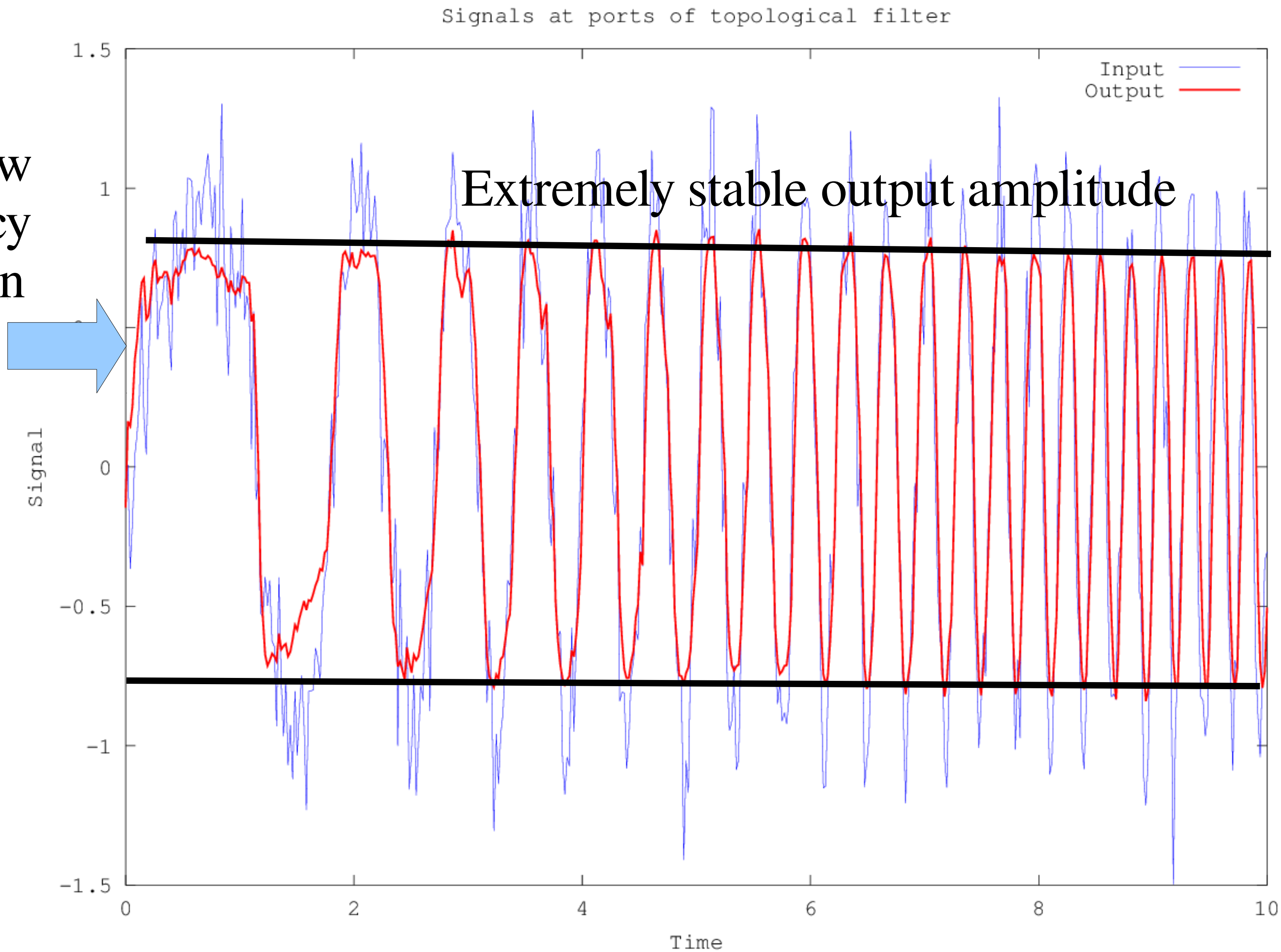
# How is this a topological filter?



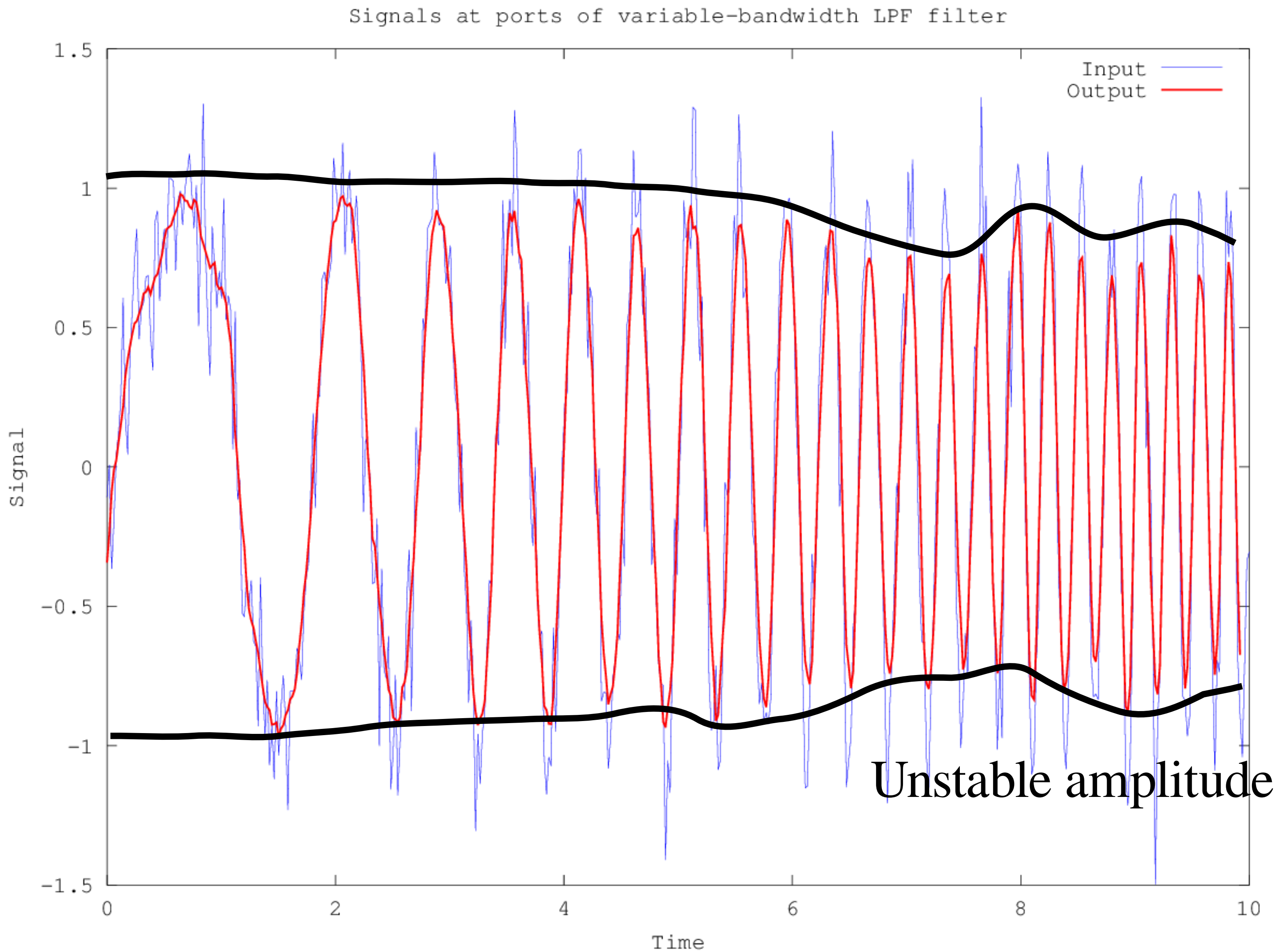
# QPLPF results

Some low frequency distortion

Extremely stable output amplitude

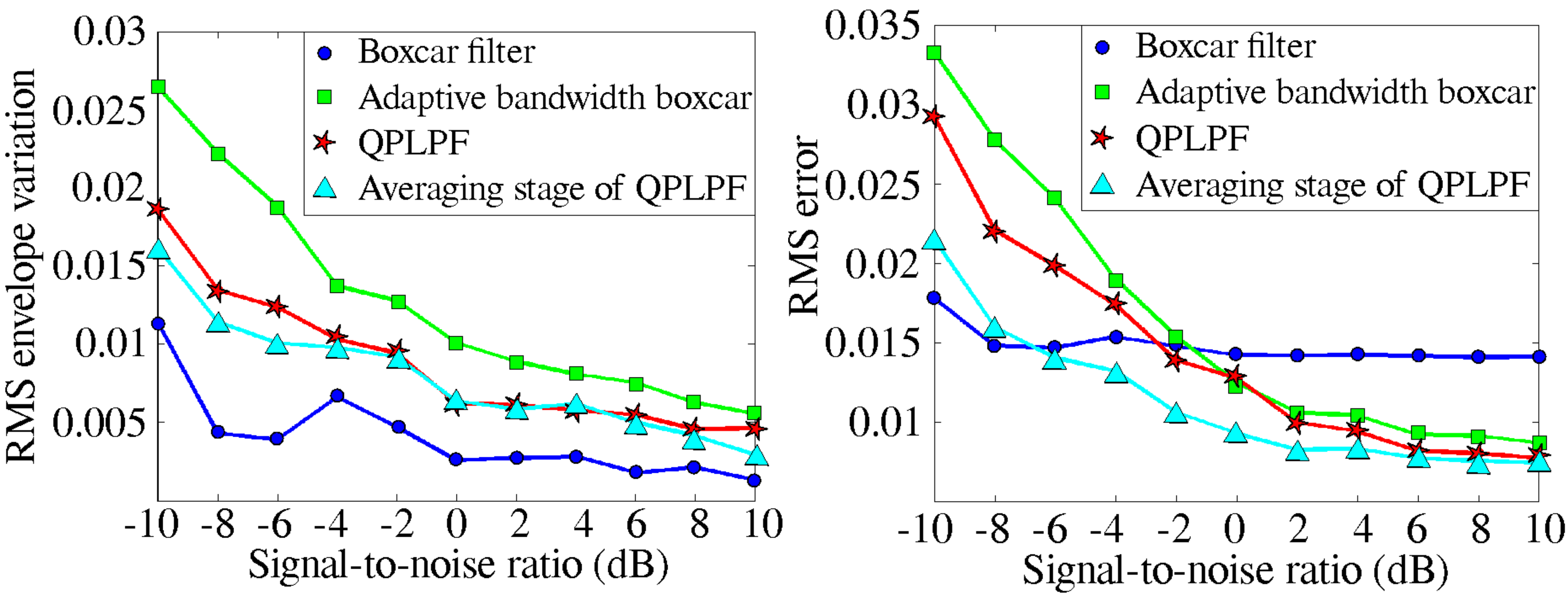


# Compare: standard adaptive filter



# Filter performance comparison

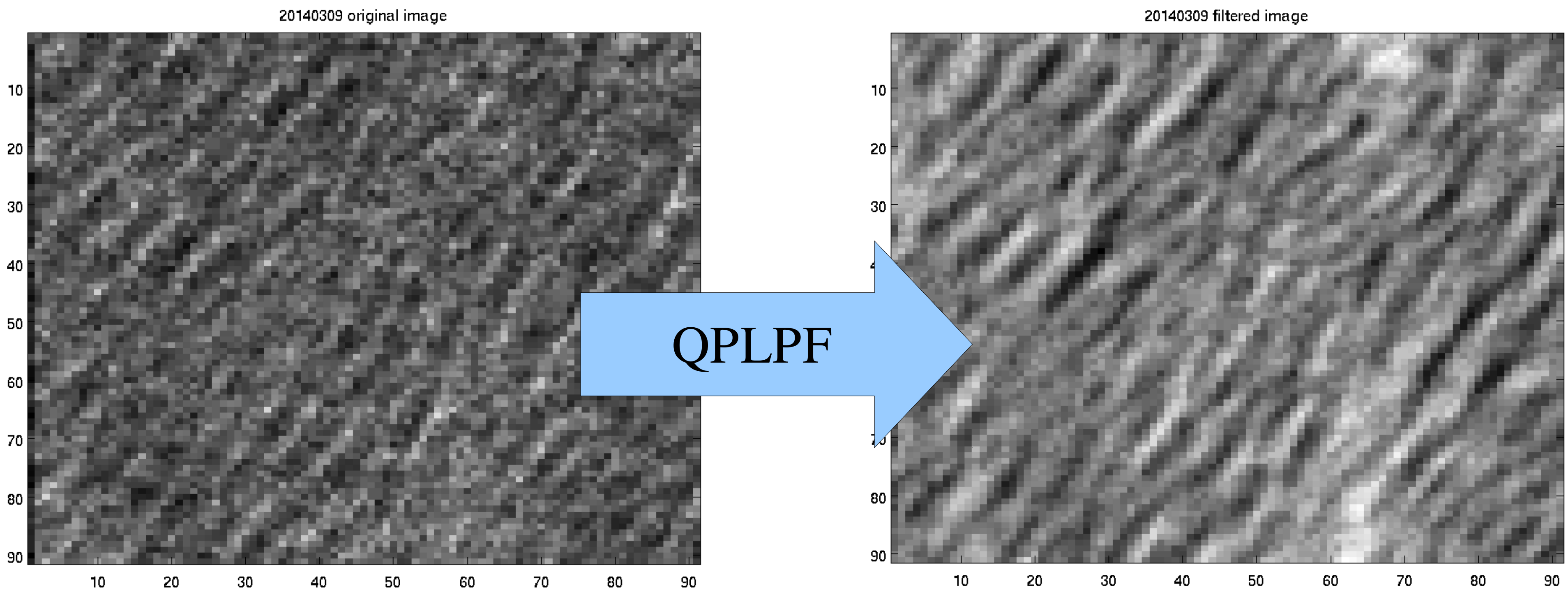
- QPLPF combines good noise removal with signal envelope stability



# Ocean radar image despeckling

After topological filtering:

- Speckle and contrast improved



# High-pass filtering

*Detecting missing and spurious data*

joint work with Fernando Benadon and Andy McGraw

# Context: Afro-Cuban drumming

photo credit:
Andrew McGraw

- Five instrumentalists
- No musical score
- Varying degrees of structure

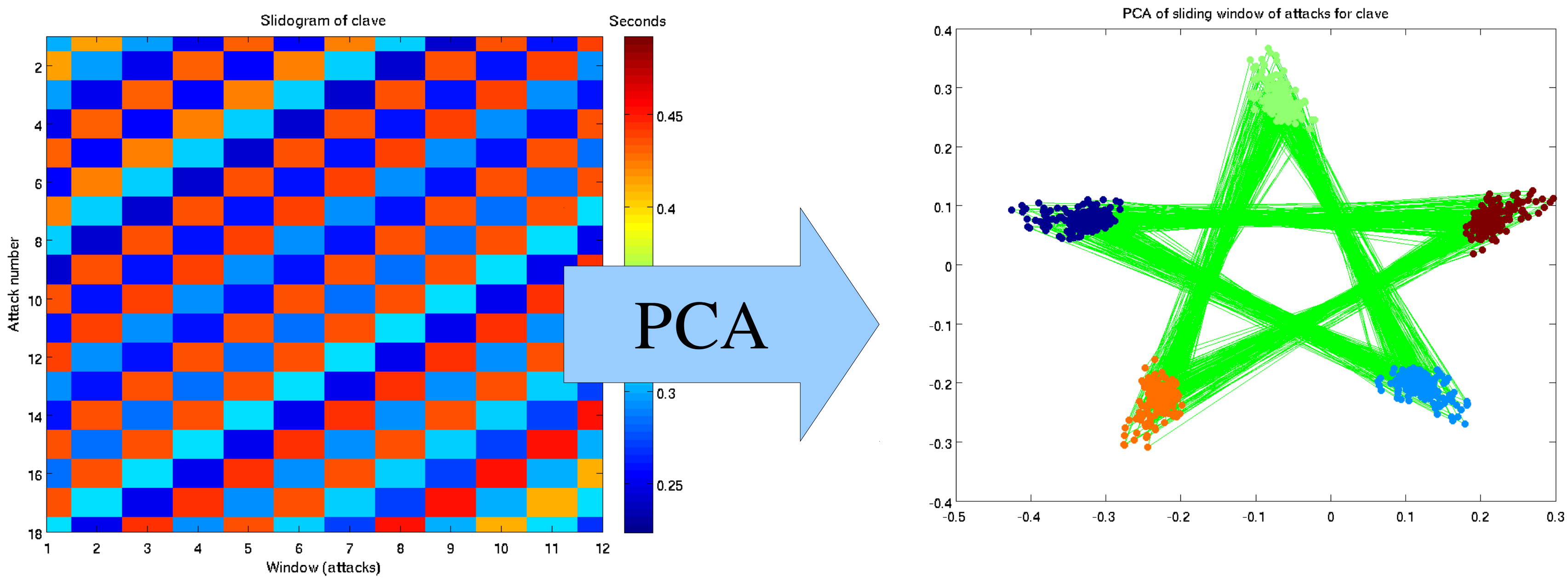
Onset list

Inter-Onset Intervals

| | | |
|---|---|---|
| 7 | [illegible] | [illegible] |
| 8 | 27.919333333 | 0.240666 |
| 9 | 28.16 | 0.064000 |
| 10 | 28.224 | 0.222666 |
| 11 | 28.446666666 | 0.124000 |
| 12 | 28.570666666 | 0.186666 |
| 13 | 28.757333333 | 0.117333 |
| 14 | 28.874666666 | 0.193333 |
| 15 | 29.068 | 0.233333 |
| 16 | 29.301333333 | 0.122666 |
| 17 | 29.424 | 0.216000 |
| 18 | 29.64 | 0.231333 |
| 19 | 29.871333333 | 0.099333 |

# Extracting musical structure

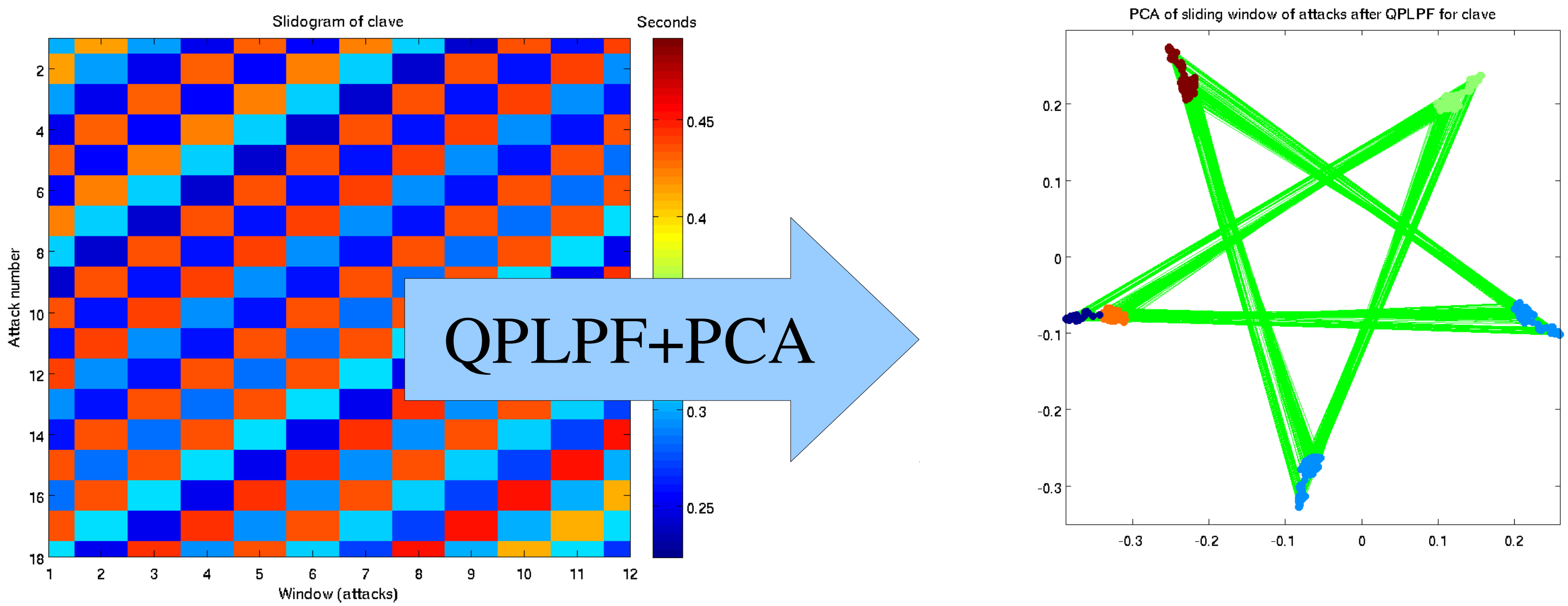
- The *clave* is highly regular… it provides the timing for the ensemble




Sliding window array

# Extracting musical structure

- The *clave* is highly regular…
- QPLPF acts by tightening the note clusters

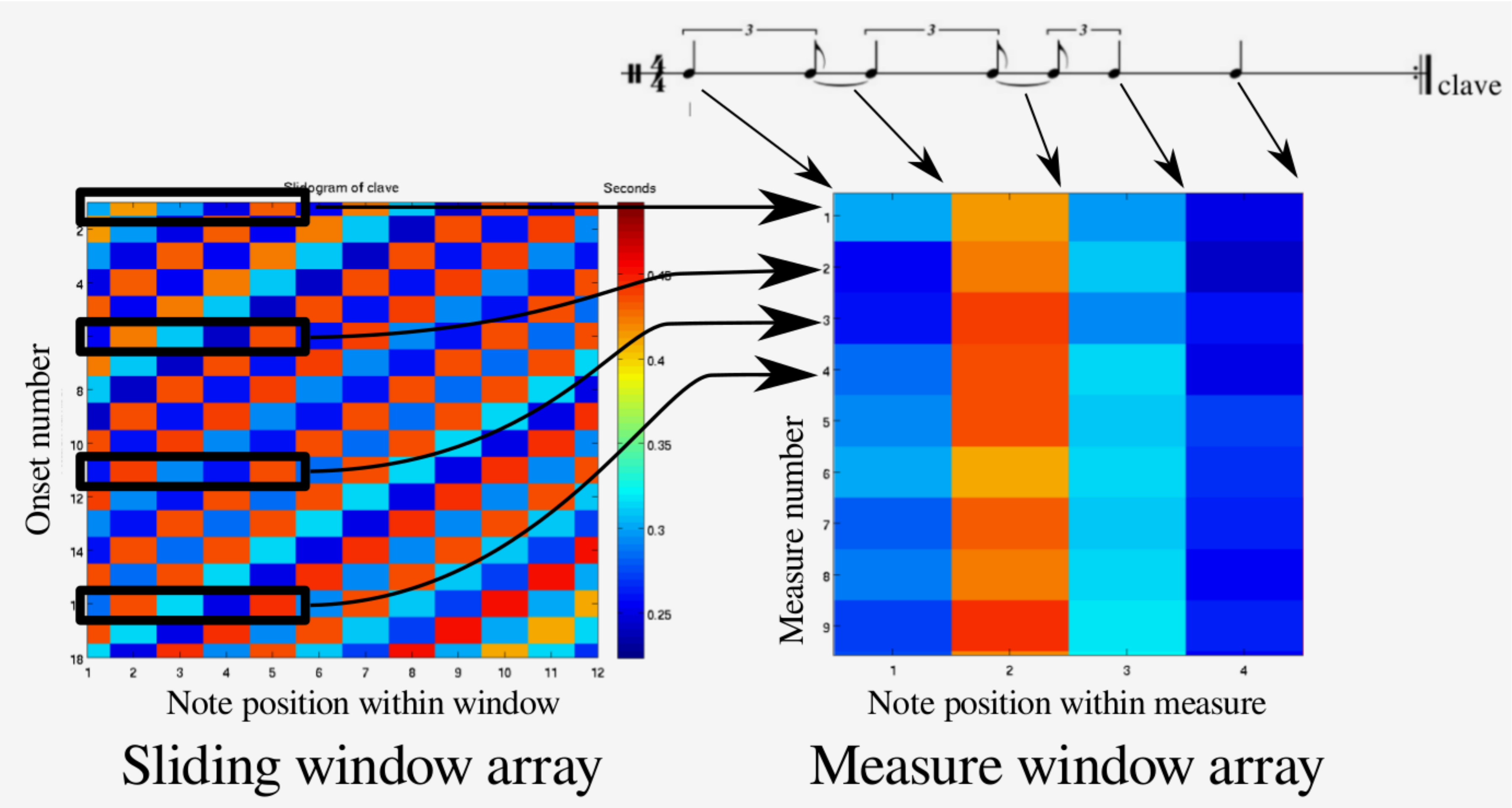



Sliding window array

(ignore the nuisance rotation!)

# Extracting musical structure

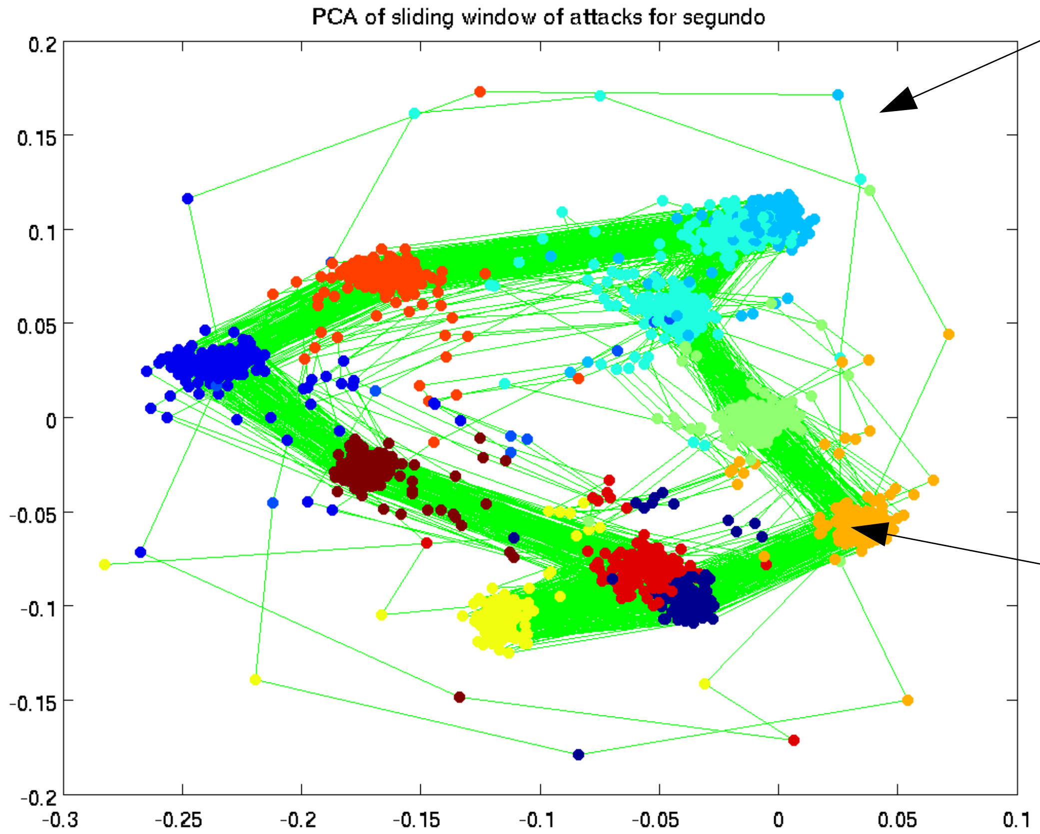
- … so much that it can be transcribed easily



# Some instruments are less clear

- The *segundo* is pretty structured...

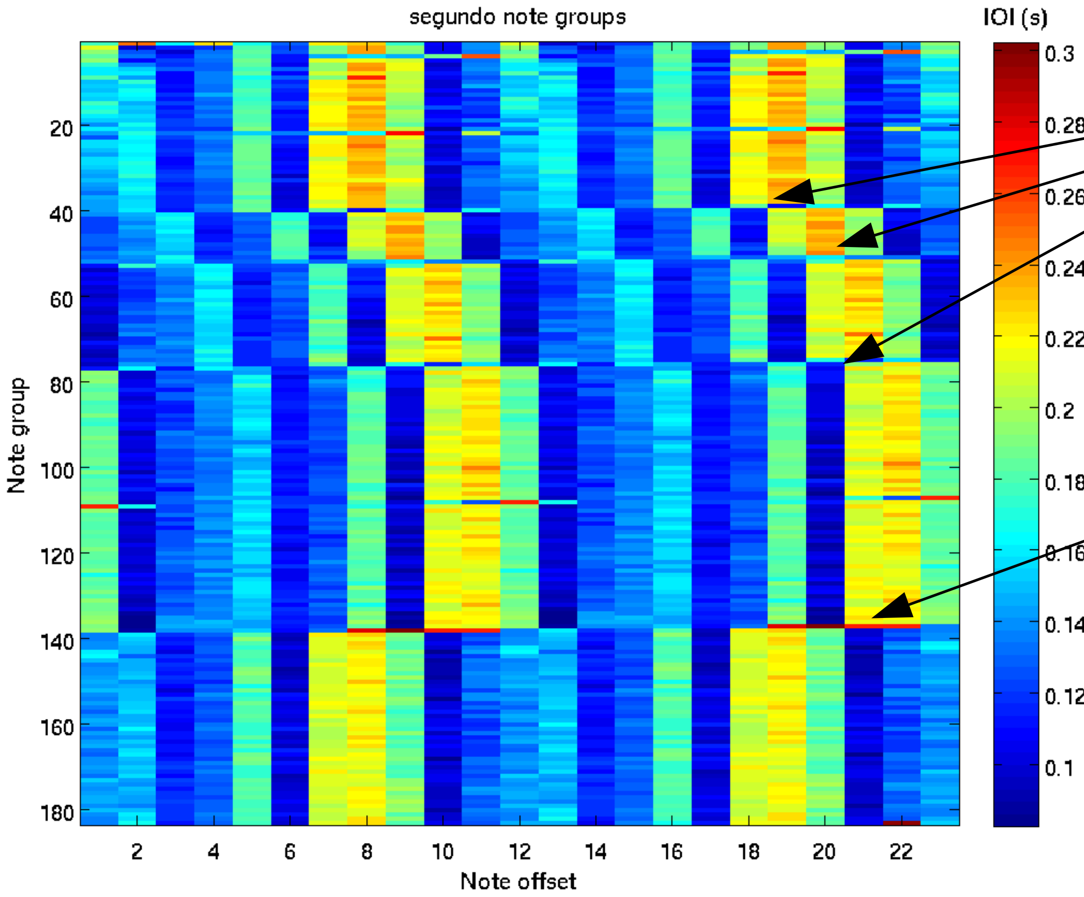
Outliers present




Note clusters from main theme

# Some instruments are less clear

- … but automated transcription is frustrated by *ghost notes*.  (There's considerable musical nuance)



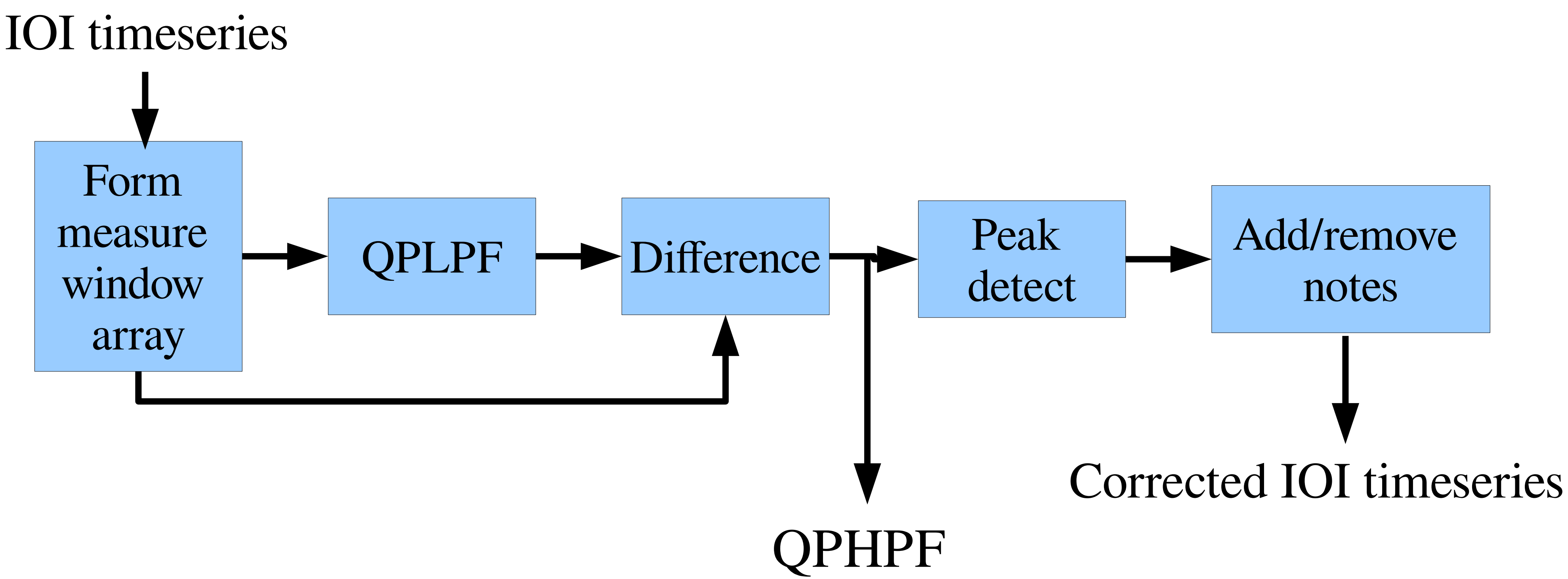

"Extra" notes

"Missing" notes

NB: These might be "extra" or "missing" on purpose!

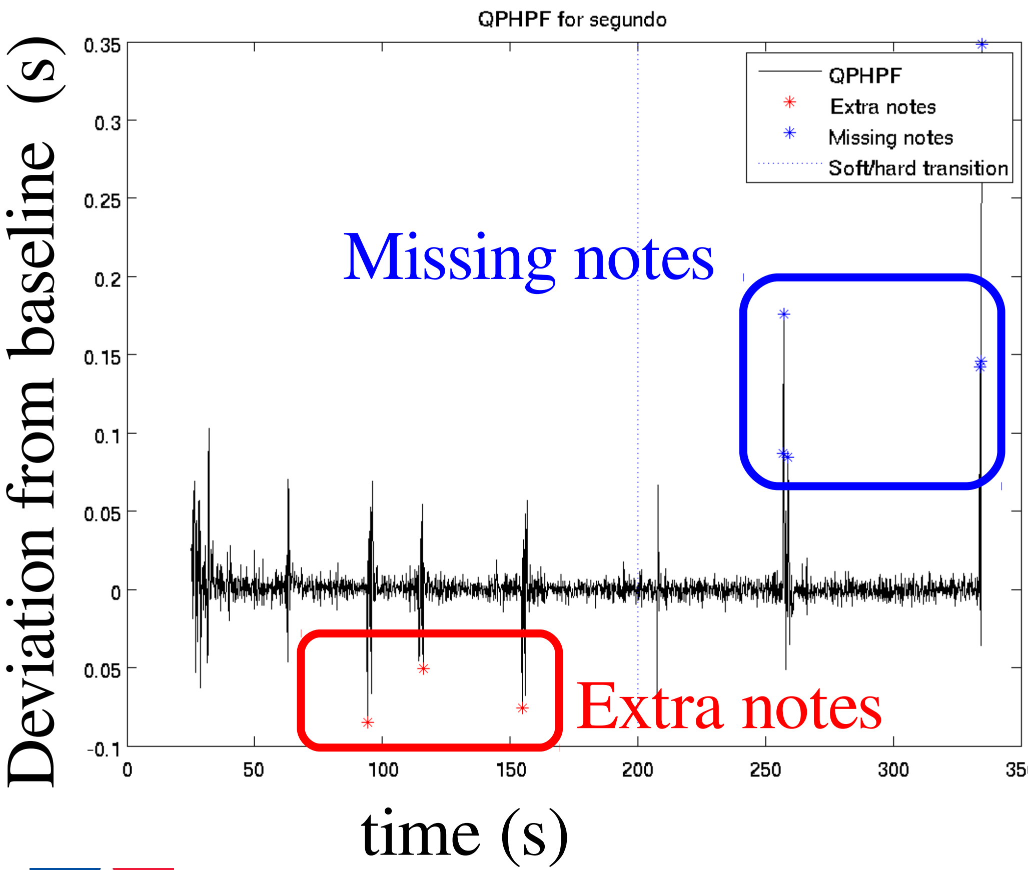
# Deghosting process

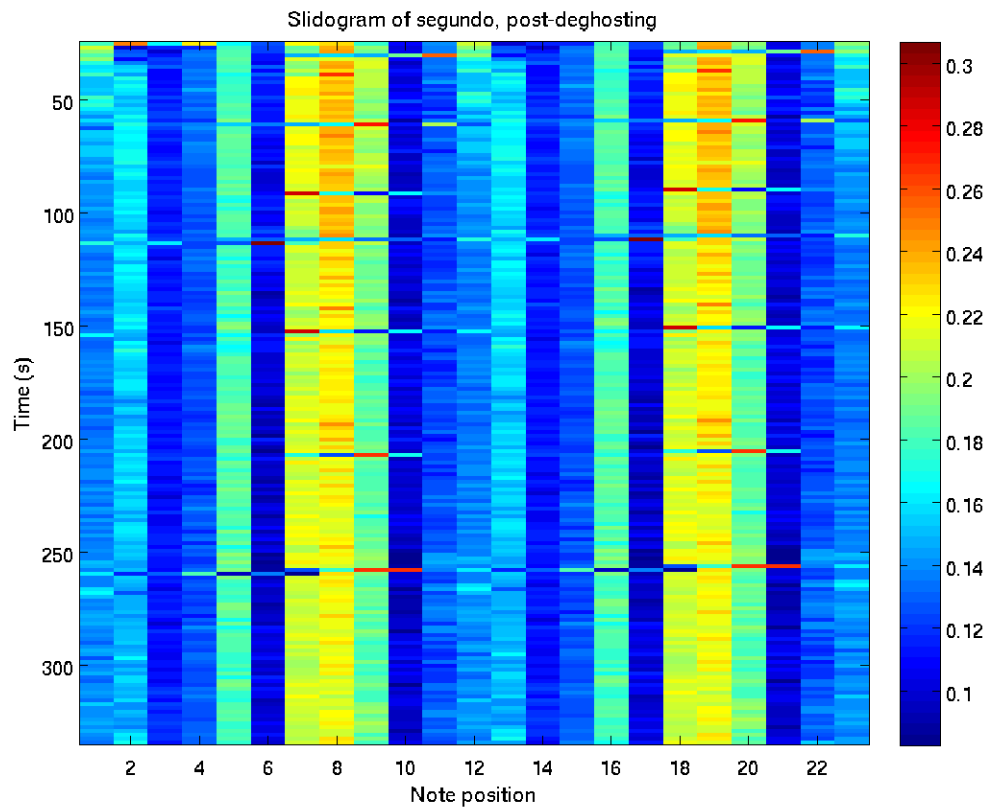
- Use QPLPF as a baseline, look at the difference!
- This is the *QuasiPeriodic High Pass Filter*



# Peak detection subtlety

- Two musically-separate halves of the piece.
- They need to be handled differently

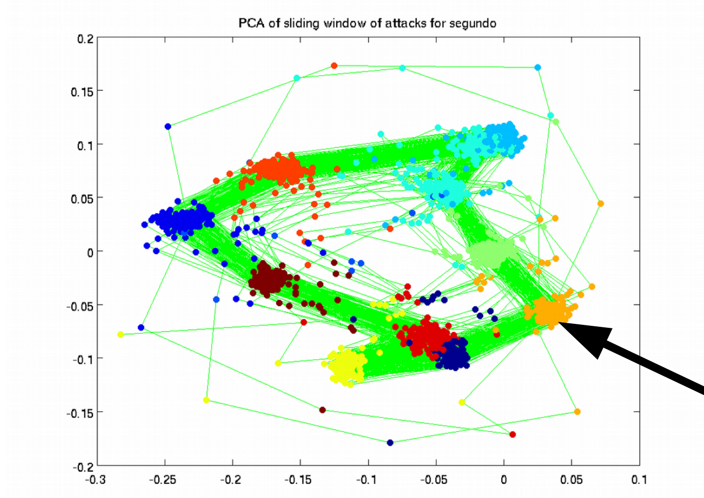


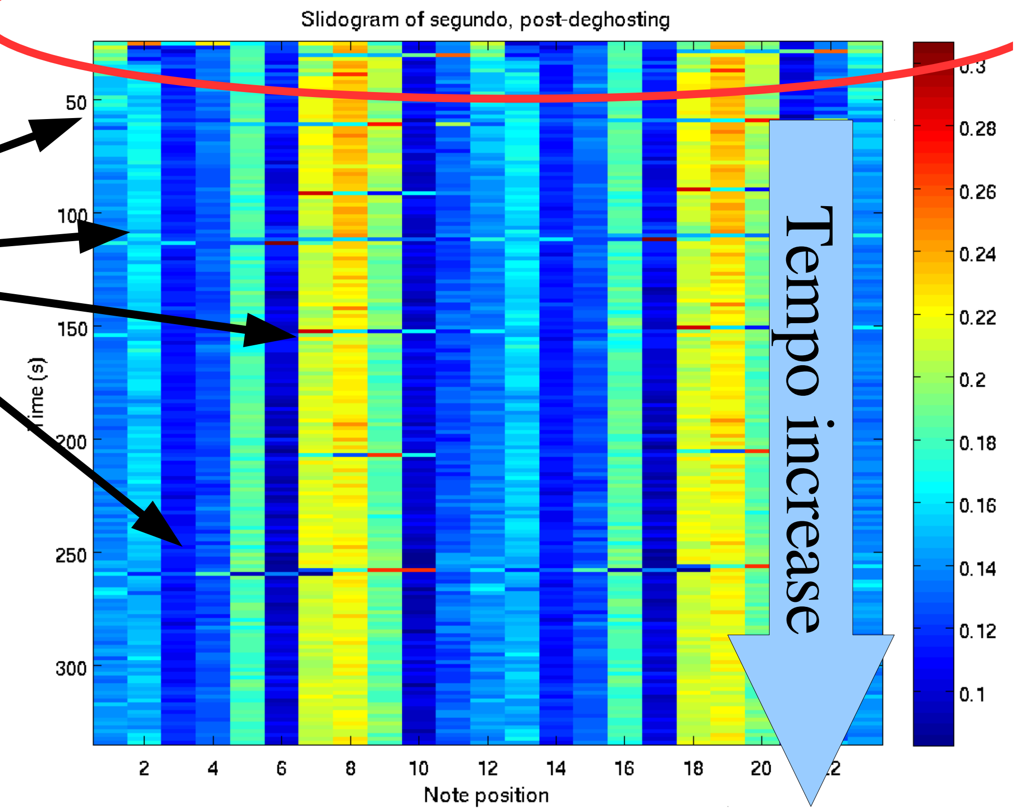



# Features now visible

First few measures are different, before stabilizing to regular pattern

Distinct anomalous measures, possibly to re-synch with other drummers, or maybe just weaker ghosts...

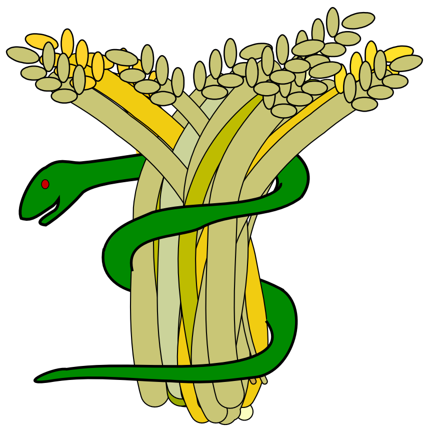



*Segundo* follows an 11-note pattern

# The future

- Computational sheaf theory
  - Small examples can be put together *ad hoc*
  - Larger ones require a software library
- `PySheaf`: a software library for sheaves
  - https://github.com/kb1dds/pysheaf
  - Includes several examples you can play with!
- Connections to statistical models need to be explored
- Extensive testing on various datasets and scenarios



# For more information

Michael Robinson

michaelr@american.edu

Preprints available from my website:

http://www.drmichaelrobinson.net/