Deep Learning in Music Informatics

Demystifying the Dark Art, Part III – Practicum

Eric J. Humphrey
04 November 2013

MARL

ismir
CURITIBA • BRAZIL

Outline

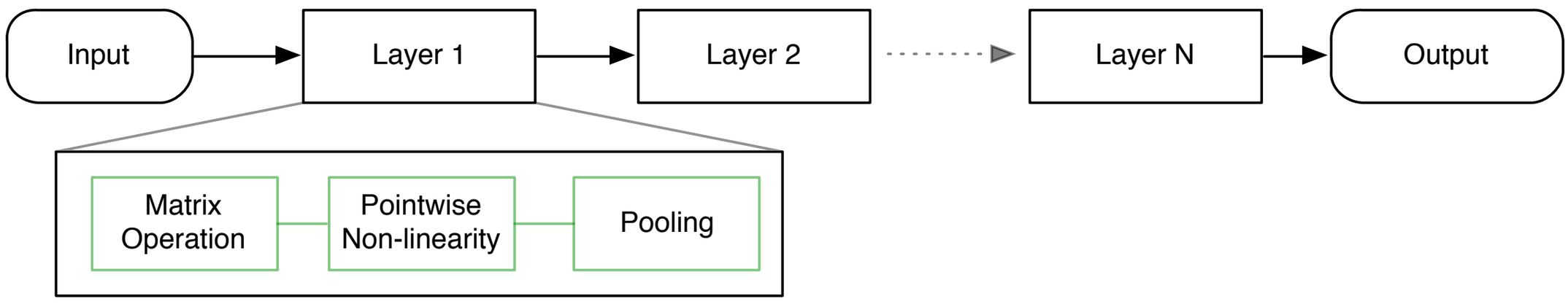
- In this part of the talk, we'll touch on the following:
  - Recap: What "is" deep learning
  - How you are already doing it (kinda)
  - Why should you consider it
  - Some tips, tricks, insight, and advice
  - How you might do it intentionally
  - A few thoughts for the future

MARL

ismir
Curitiba • Brazil

Deep learning is...

- Cascade of multiple layers, composed of a few simple operations
  - Linear algebra
  - Point-wise nonlinearities
  - Pooling



MARL

ismir
Curitiba • Brazil

Nonlinearites enable complexity

- Cascaded non-linearities allow for complex systems composed of simple, linear parts
  - The composite of two linear systems is just another linear system
  - The composite of two non-linear systems is an entirely different system

$$y = B(Ax) = (BA)x$$

linear

$$y = h(Bh(Ax)) \neq h(BAh(x))$$

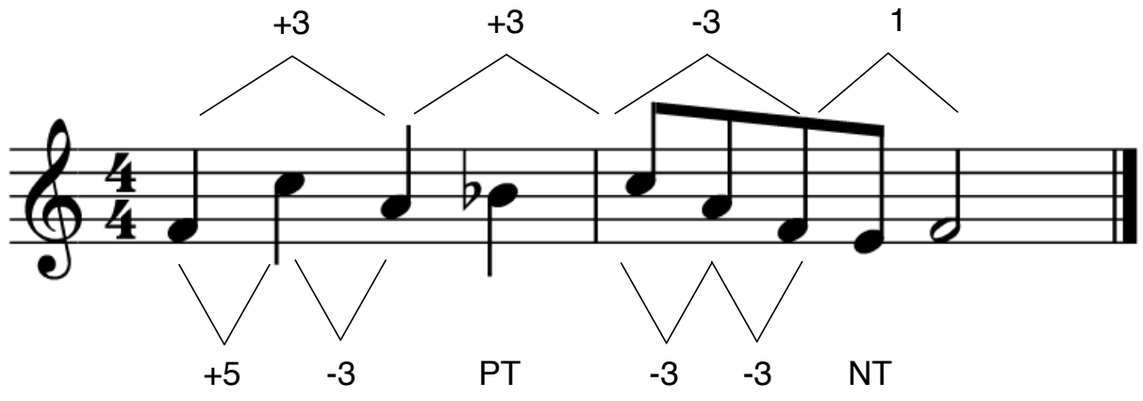
nonlinear

MARL

ismir
CURITIBA • BRAZIL

Why is this relevant to music?

- Quite literally, music is composed!
  - Hierarchies of pitch and loudness form chords and melodies, phrases and sections, eventually building entirely pieces.
  - Deep structures are well suited to encode these relationships.



MARL

ismir
Curitiba • Brazil

Outline

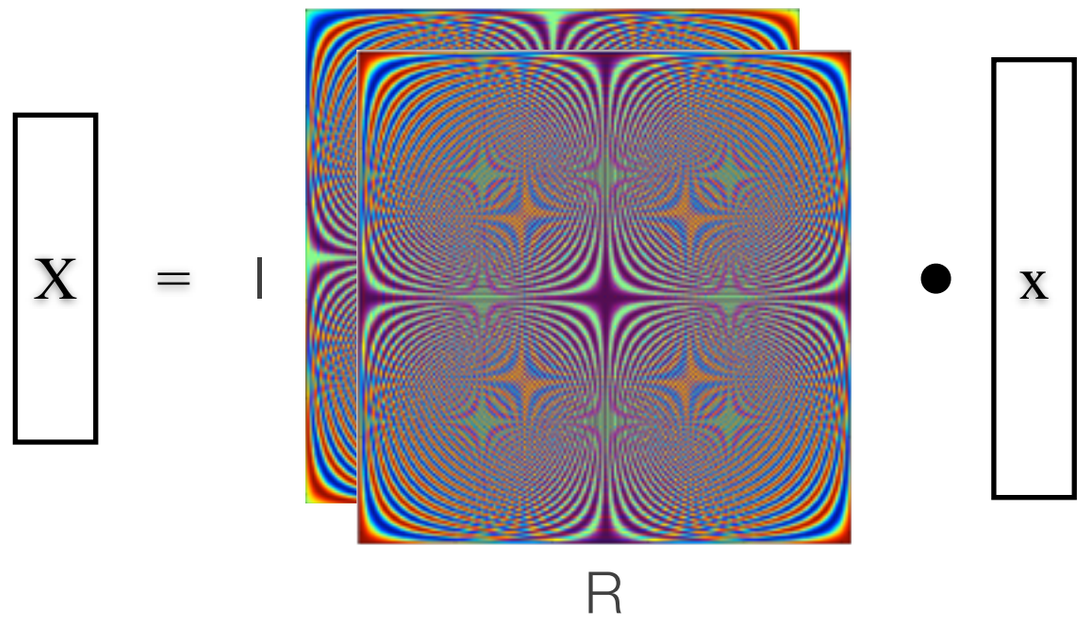
- In this part of the talk, we'll touch on the following:
  - Recap: What "is" deep learning
  - How you are already doing it (kinda)
  - Why should you consider it
  - Some tips, tricks, insight, and advice
  - How you might do it intentionally
  - A few thoughts for the future

MARL

ismir
Curitiba • Brazil

The ever versatile DFT

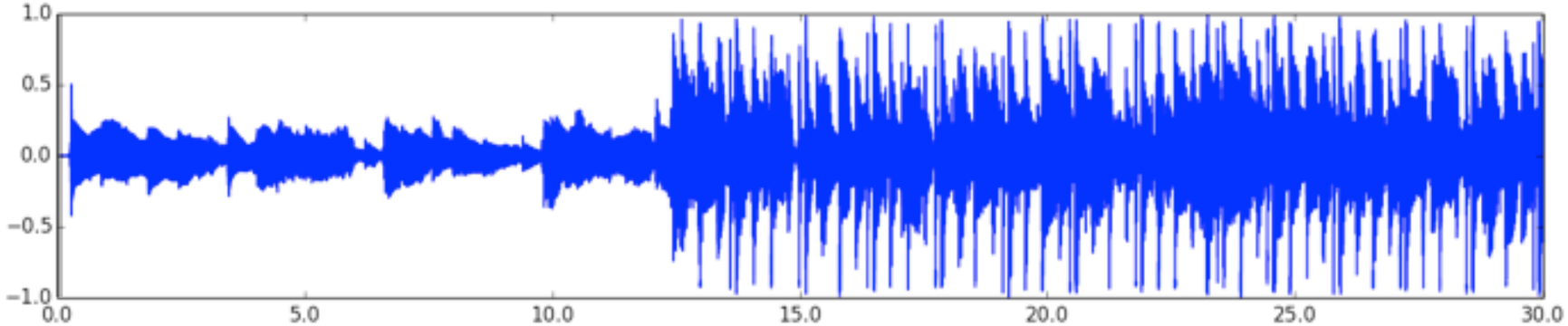
$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N}$$

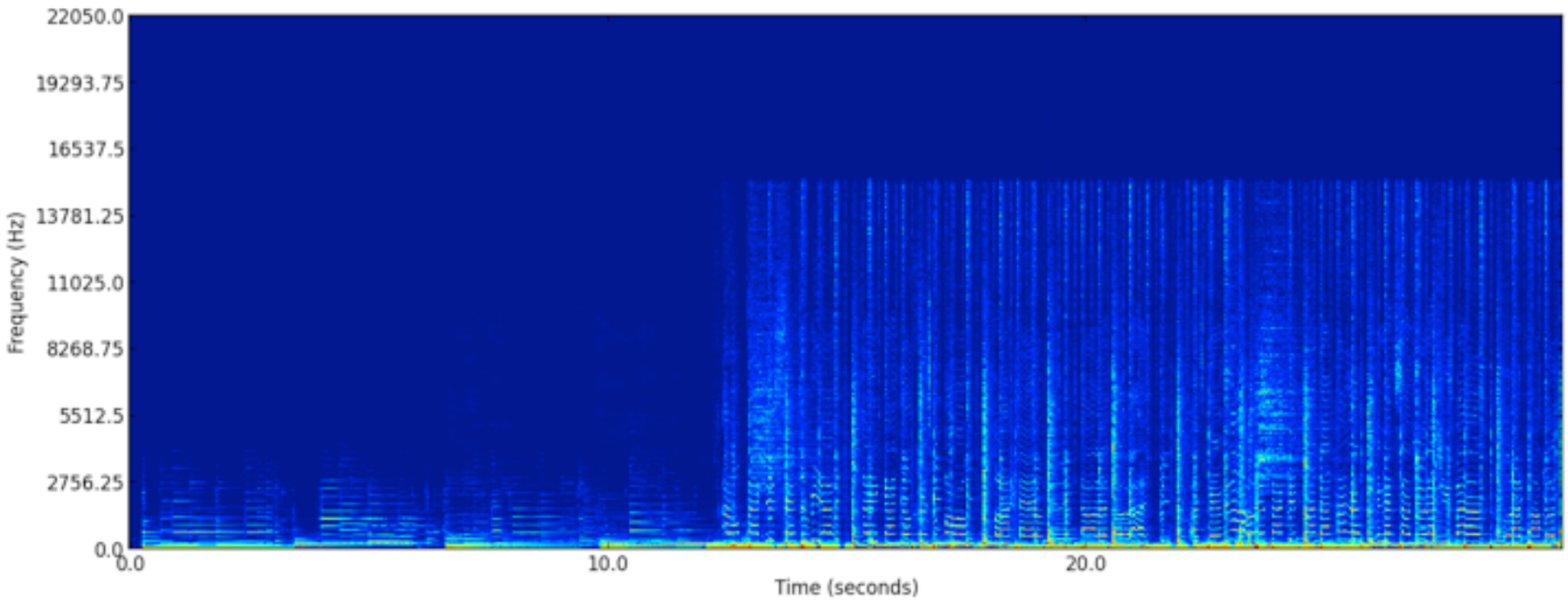


MARL

ismir
CURITIBA • BRAZIL

Short-Time Fourier Transform





MARL

ismir
Curitiba • Brazil

Some common MIR operations

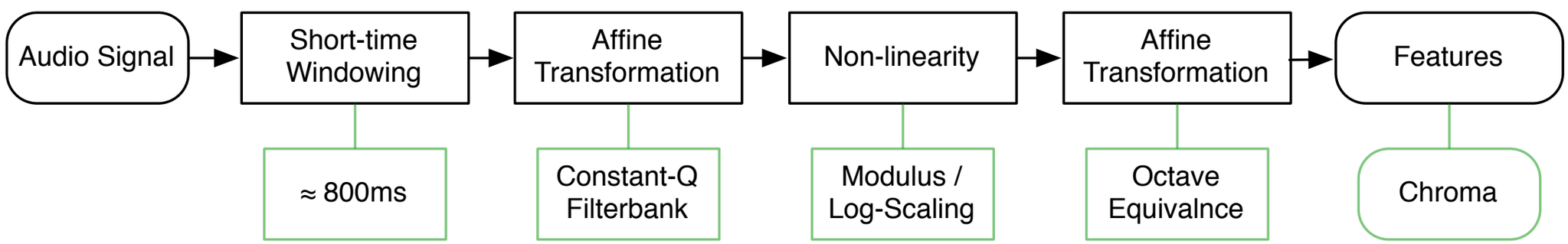
- Linear algebra
  - The DFT is a general affine transformation (dot-product)
    - ...followed by an absolute value (full wave rectification)
    - ...followed by a logarithm
  - The DCT is a general linear affine transformation
  - PCA is a learned, linear affine transformation
  - NMF is a learned, linear affine transformation
- Non-linearities:
  - Half/Full-wave rectification, peak picking, logarithms
- Pooling: Histograms, standard deviation, min/max/median

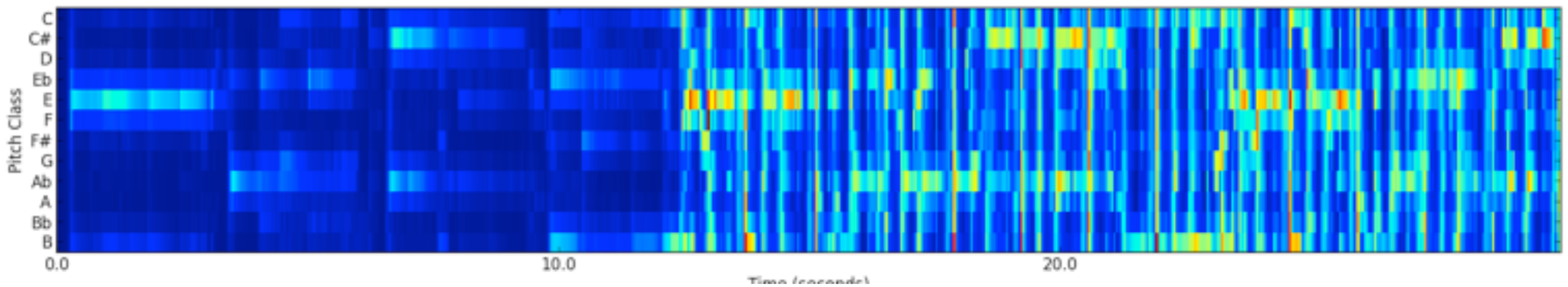
The pieces of deep learning are everywhere in feature design.

MARL

ismir
CURITIBA • BRAZIL

Chroma

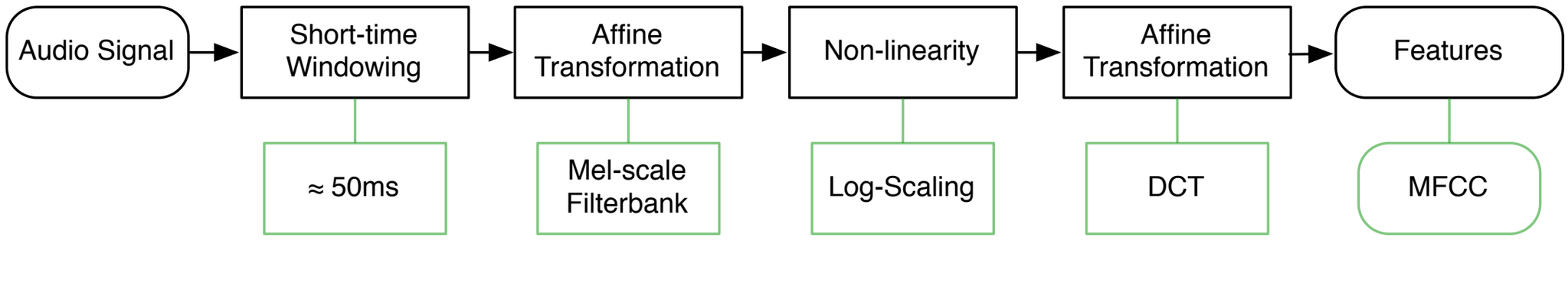


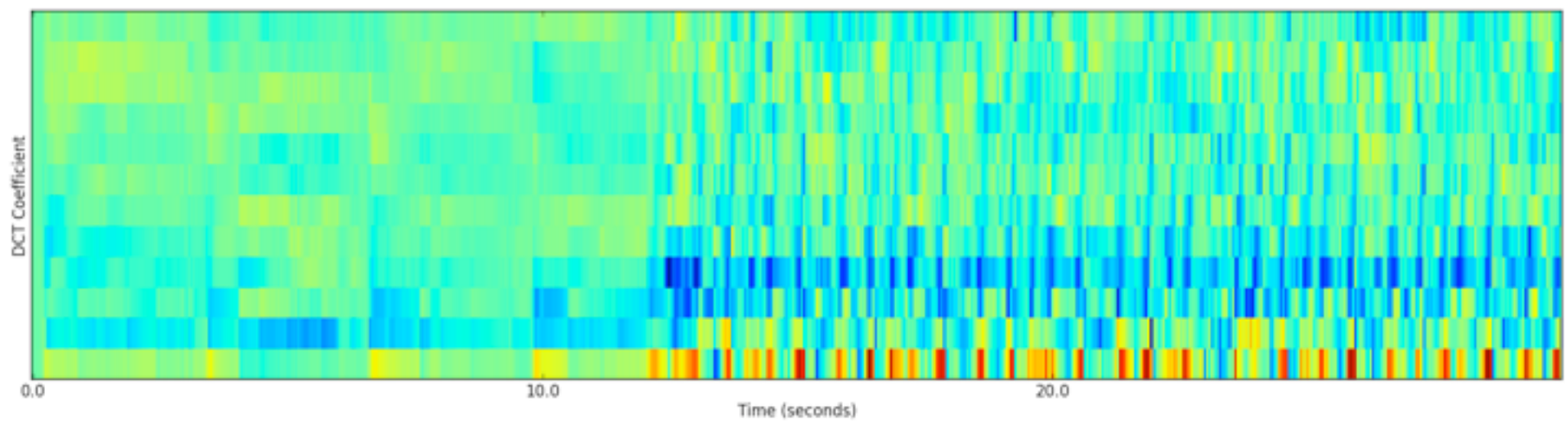


MARL

ismir
CURITIBA • BRAZIL

MFCCs





MARL

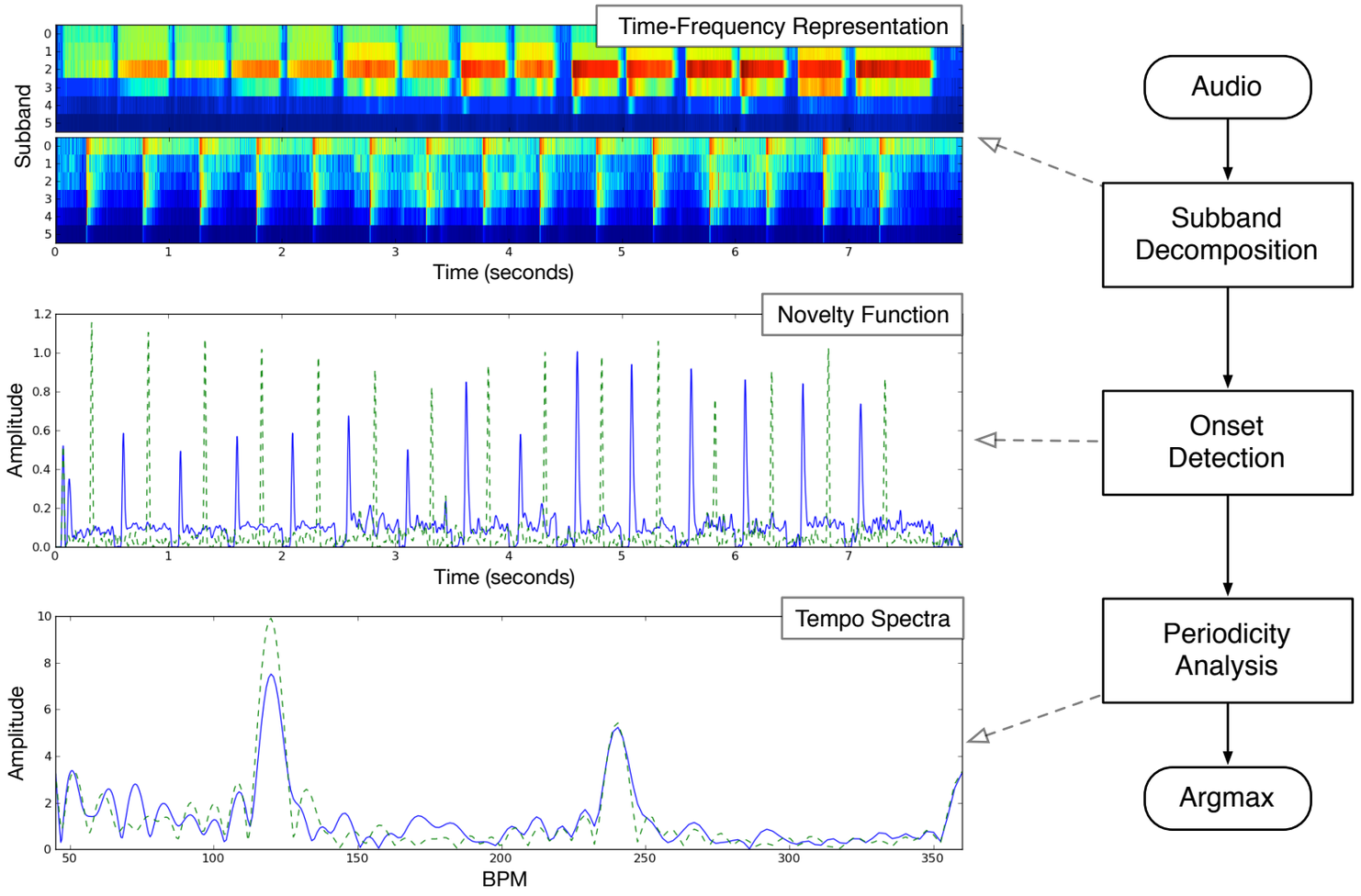
ismir
Curitiba • Brazil

Feature design is based on a shared intuition:
Build invariance into your representations.

MARL

ismir
CURITIBA • BRAZIL

Case in Point: Tempo Estimation



MARL

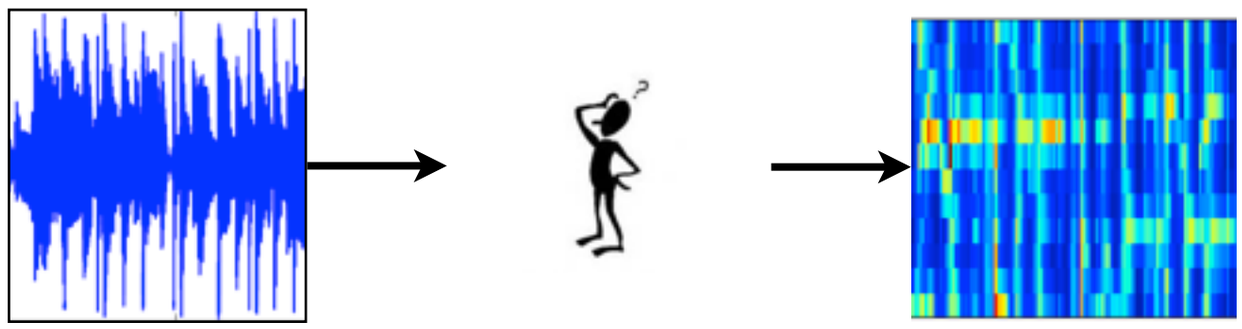
ismir
Curitiba • Brazil

Outline

- In this part of the talk, we'll touch on the following:
  - Recap: What "is" deep learning
  - How you are already doing it (kinda)
  - Why should you consider it
  - Some tips, tricks, insight, and advice
  - How you might do it intentionally
  - A few thoughts for the future

The goal of feature extraction

- Model the relationship between inputs (x) and observations (y)
- Restated: Develop representations that encode some desired invariance
  - Robust when this is captured
  - Noisy when variance is uninformative / misleading
- Why is this difficult?
  - You have to know what you want
  - You have to know how to do it

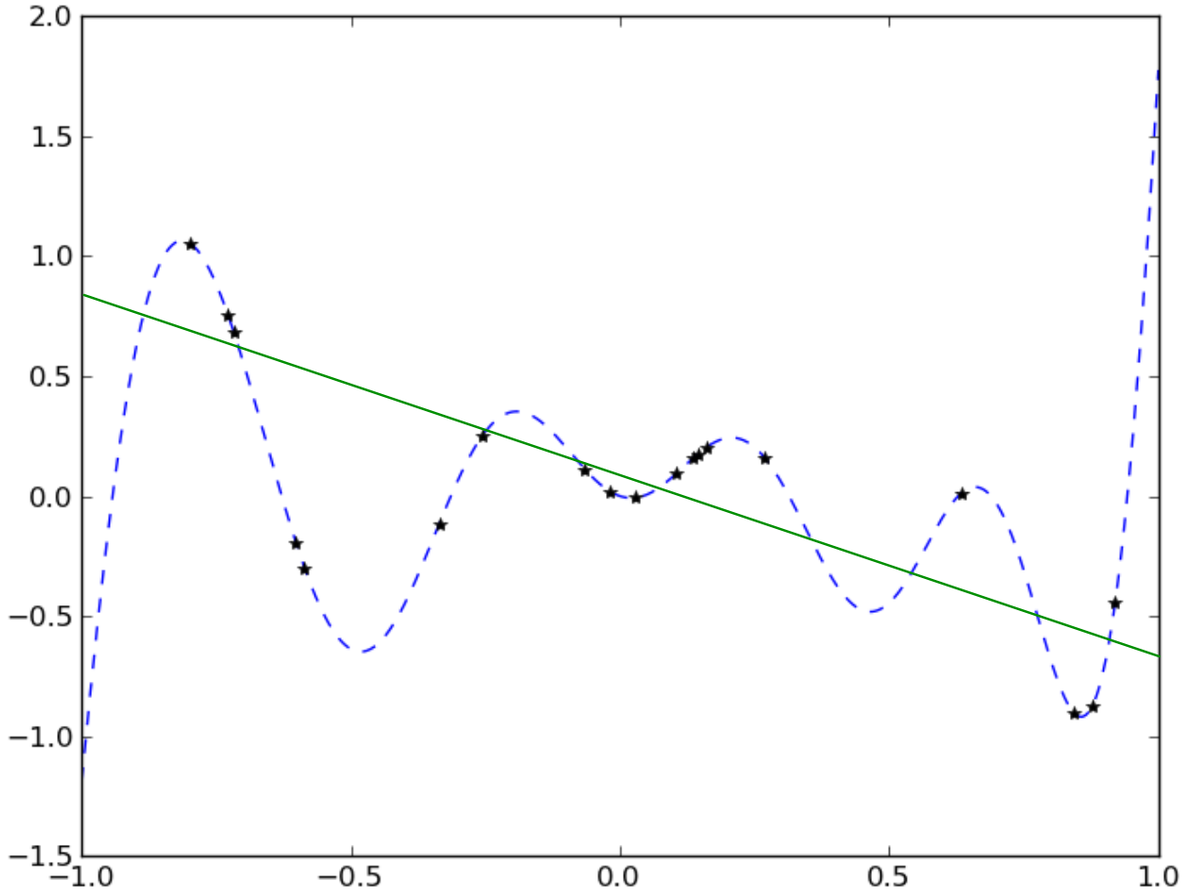


audio function chroma

MARL

ismir
CURITIBA • BRAZIL

The Simplest Function



latent function

observations

model

$$y = mx + b$$

$$\Theta = [m, b]$$

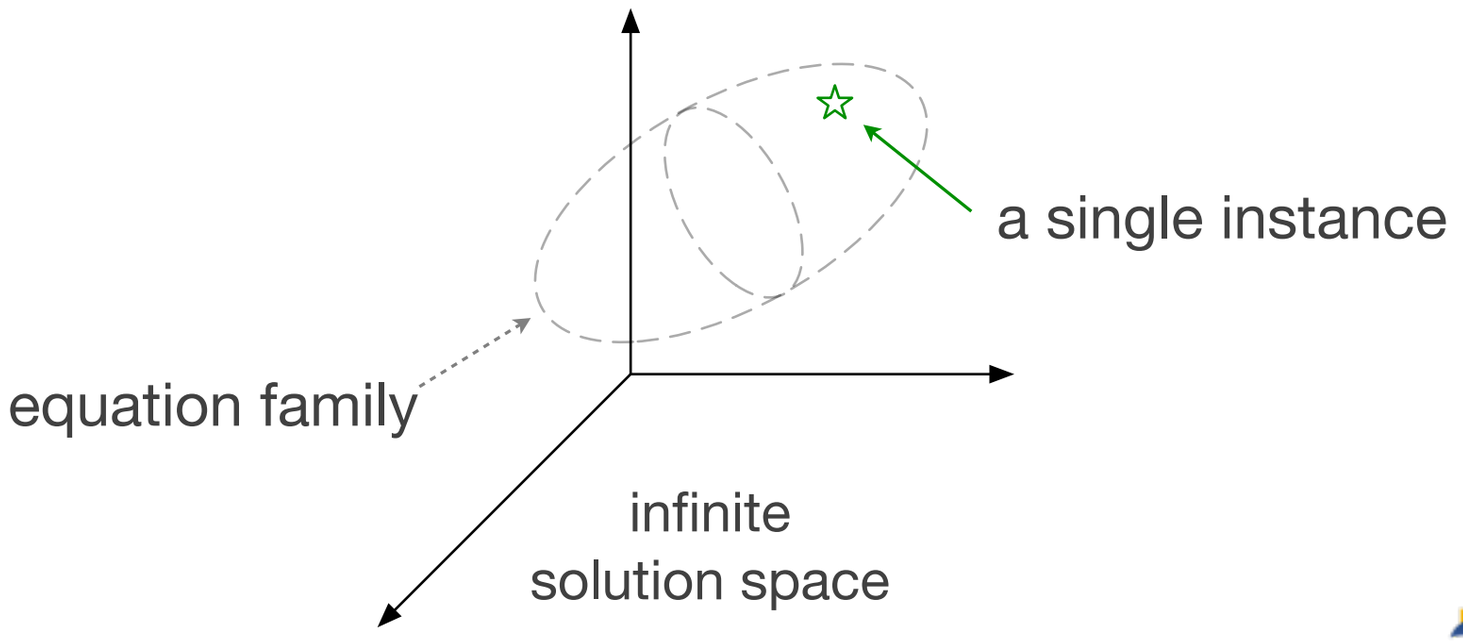
$$y = f(x|\Theta)$$

MARL

ismir
Curitiba • Brazil

Equations ≠ Parameters

- Building good functions consists of two distinct problems:
  - Getting the right equation family (general)
  - Getting the right parameterization (specific)



MARL

ismir
CURITIBA • BRAZIL

Feature design proceeds by choosing an equation family and a specific parameterization.

MARL

ismir
Curitiba • Brazil

How do we choose parameters?

- Often, manually adjust parameters to optimize some objective function
- A deep network layer is the same equation family as:
  - The DFT
  - PCA <- learned!
  - NMF <- learned!
  - ...and plenty others.

$$y = h(Wx + b)$$

- The only difference lies in the parameterization

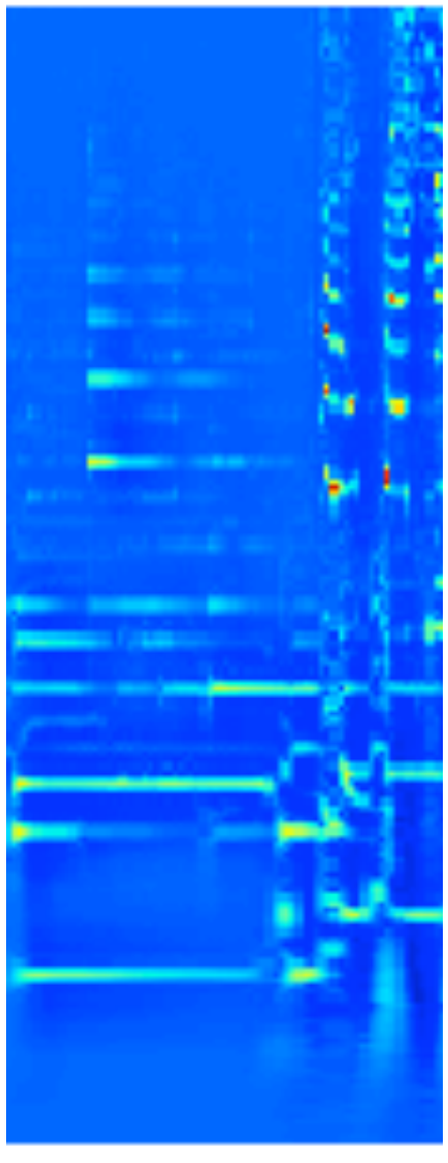
Consider Chroma

- “Pitch Class Profiles” (Fujishima, 1999)
- De facto standard harmonic representation of audio
- Several off-the-shelf implementations
- Refining chroma extraction for over a decade
- Designed for chord recognition, now used for other tasks
  - Structural analysis
  - Cover song retrieval

Chroma is octave equivalence, right?

Chroma

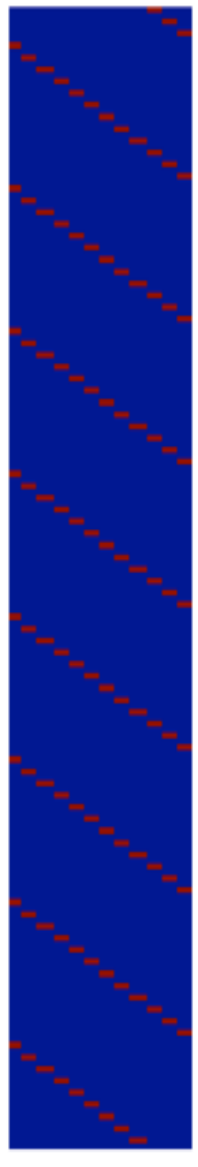
=



CQT

T

•

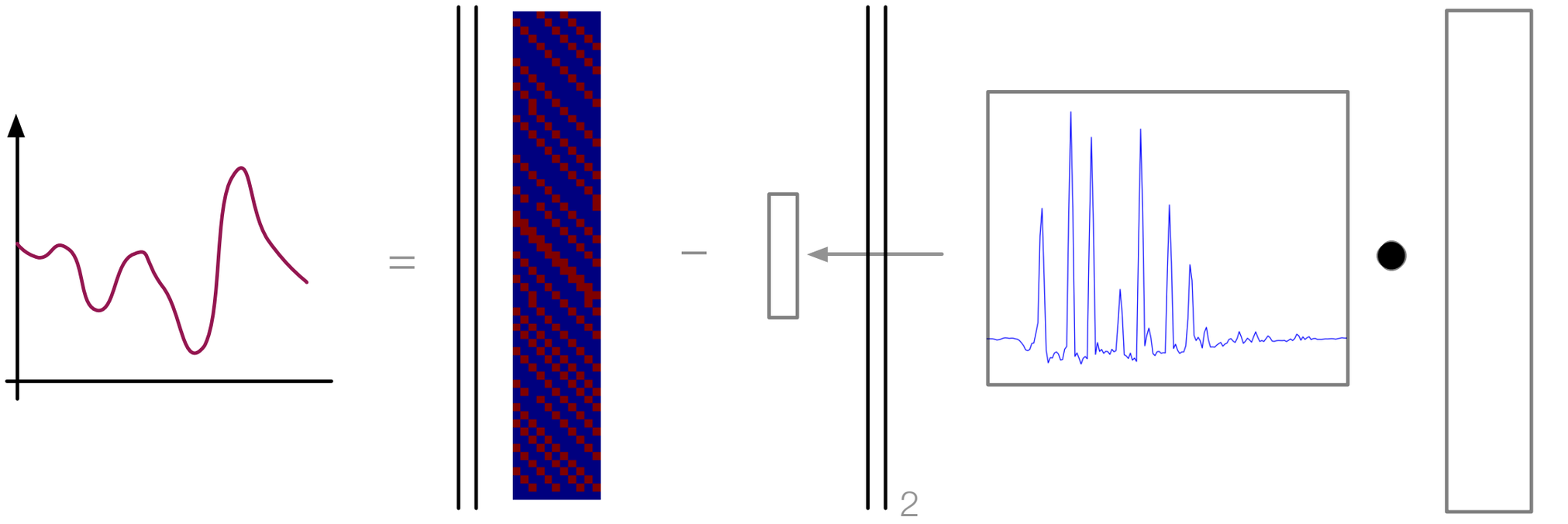


Weights

MARL

ismir
CURITIBA • BRAZIL

What if we learn these weights?



MARL

ismir
CURITIBA • BRAZIL

Deep learning is just a way of finding good parameters for the equations we already use.

MARL

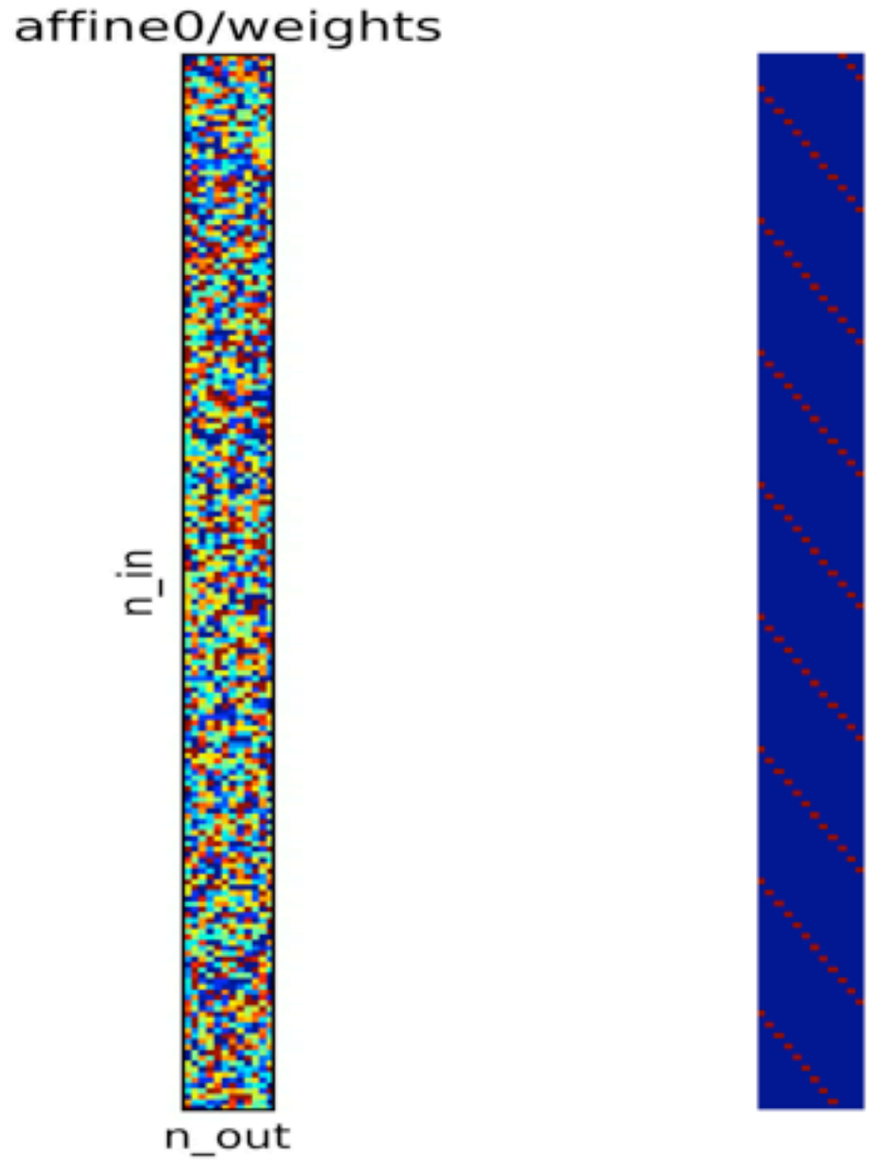
ismir
Curitiba • Brazil

...and those parameters might not be what you expected.

MARL

ismir
CURITIBA • BRAZIL

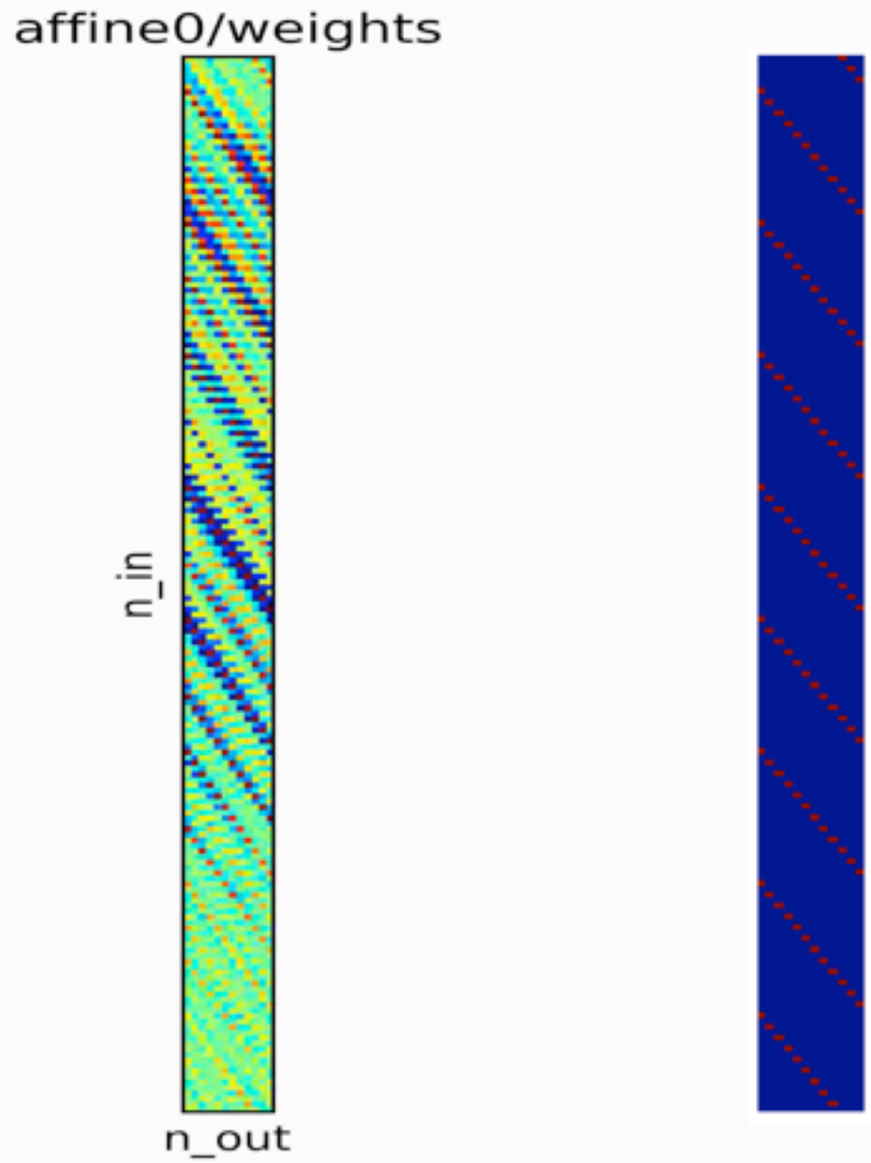
Chroma weights - Start



MARL

ismir
Curitiba • Brazil

Chroma weights - End

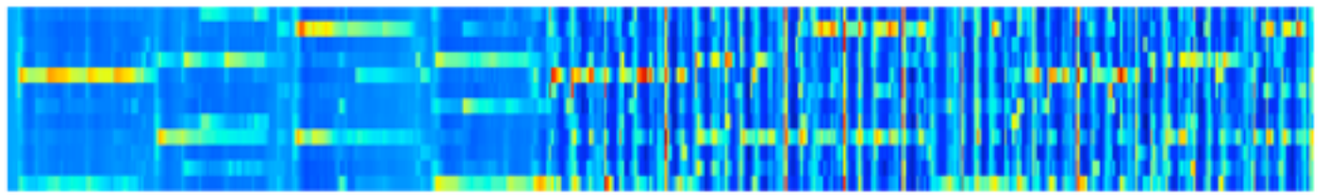


MARL

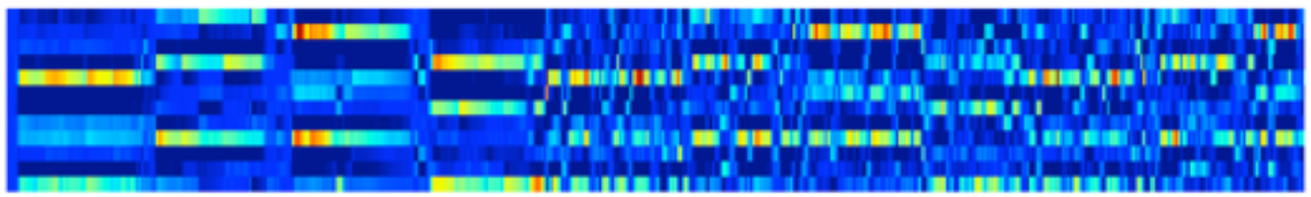
ismir
CURITIBA • BRAZIL

Chroma, Side-by-side

Octave Equivalence



Learned Transform

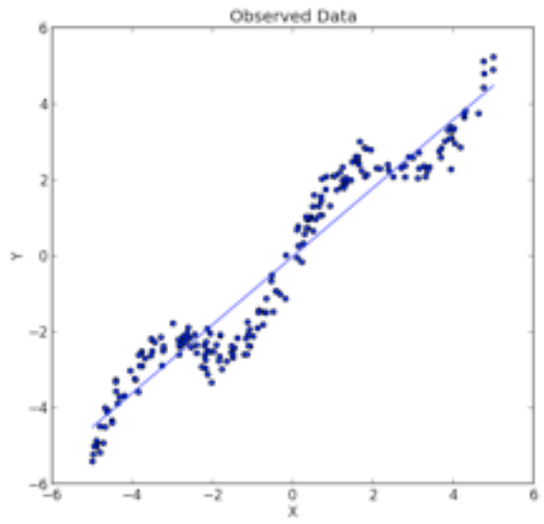


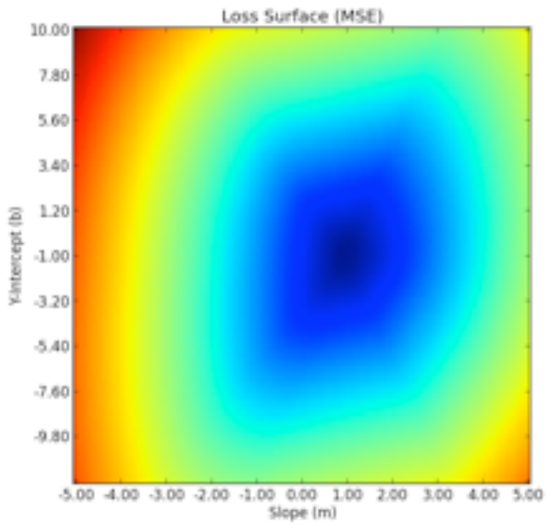
MARL

ismir
Curitiba • Brazil

Learning == Searching!

- Given an objective function, you can find, rather than choose, good parameters
- The equation family acts a constraint on the search space





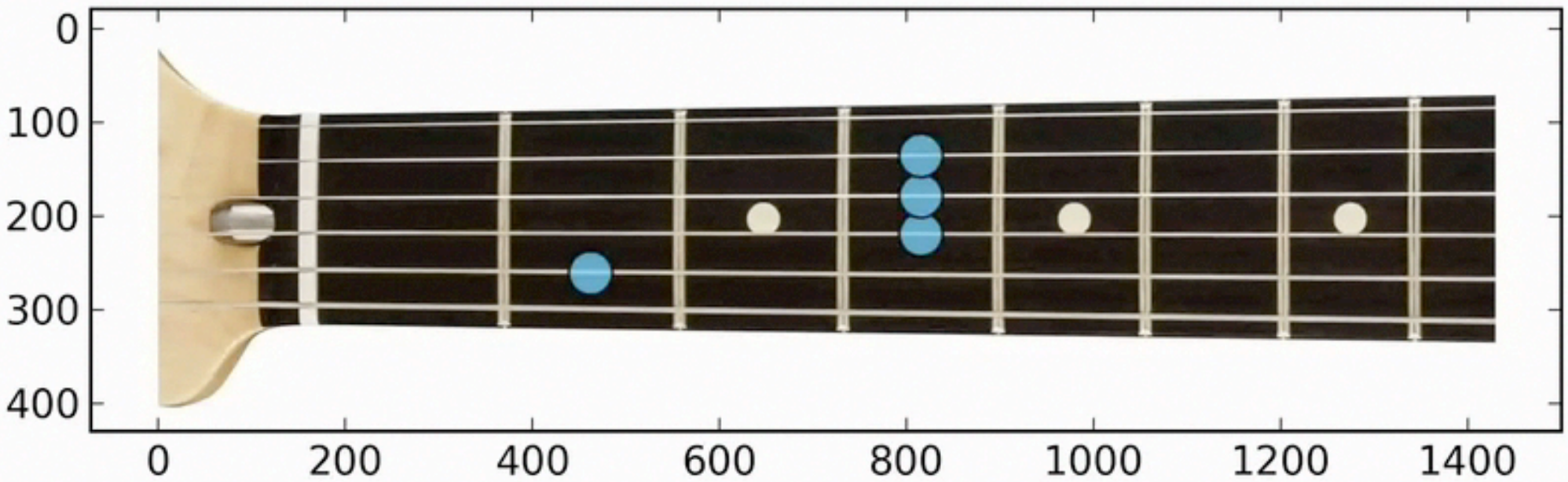
MARL

ismir
CURITIBA • BRAZIL

Feature Design *vs* Function Design

- Feature design is preferable when you have almost no data
  - Time and effort intensive process
  - Manually optimizing parameters is slow
- Function design is preferable when you have any amount of data
  - Same principles, slightly relaxed formulation
  - Quickly explore new representations you might not know how to derive

Like a fretboard, perhaps?



MARL

ismir
CURITIBA • BRAZIL

Outline

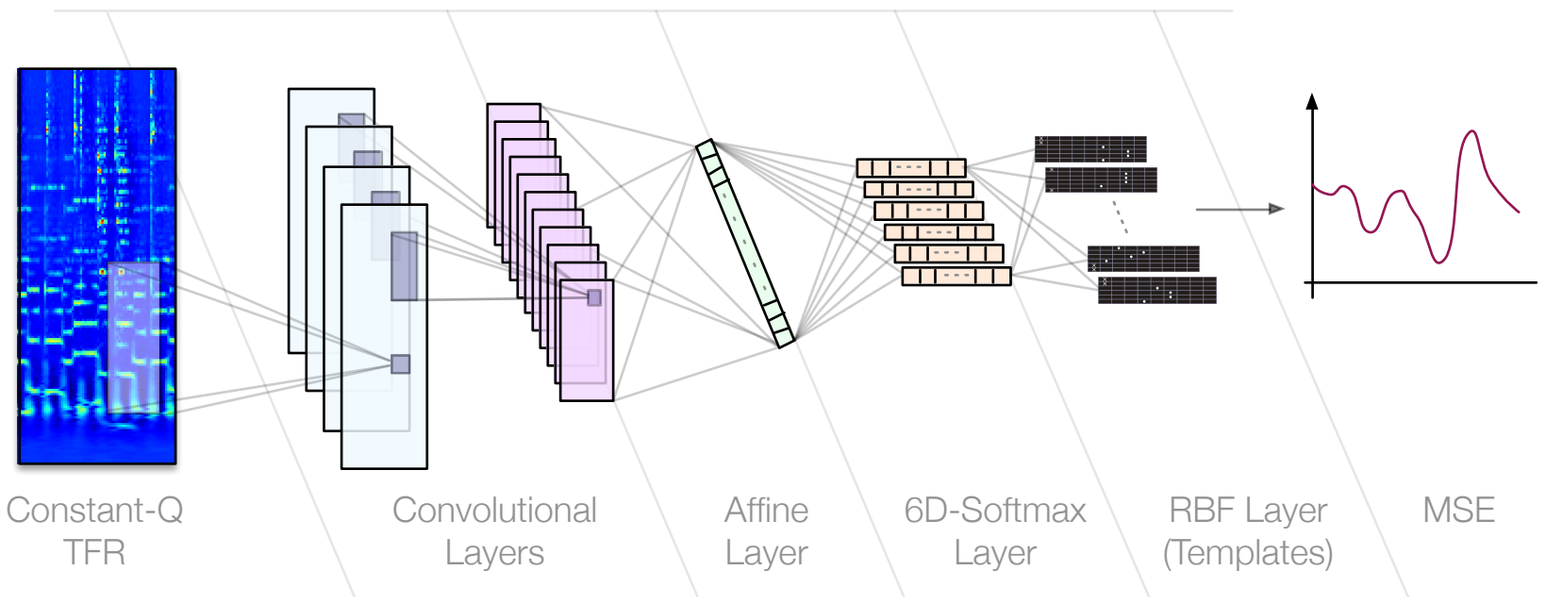
- In this part of the talk, we'll touch on the following:
  - Recap: What "is" deep learning
  - How you are already doing it (kinda)
  - Why should you consider it
  - Some tips, tricks, insight, and advice
  - How you might do it intentionally
  - A few thoughts for the future

MARL

ismir
Curitiba • Brazil

Designing deep networks

- We already kind of do this
  - How many principal components do you keep?
  - What is the window size of your DFT?
  - How many channels should your filterbank have?
- Use the same intuition to design deep networks



Designing your loss function

- Optimization criteria is extremely important
- Think of it like any greedy system (economies, children, etc)
  - Encourage and reward good behavior
  - Penalize bad behavior
- Anticipate poor local minima
- Take advantage of domain knowledge!
  - How can we steer it toward the right answer?
  - How can we use musical understanding to restrict the search space?

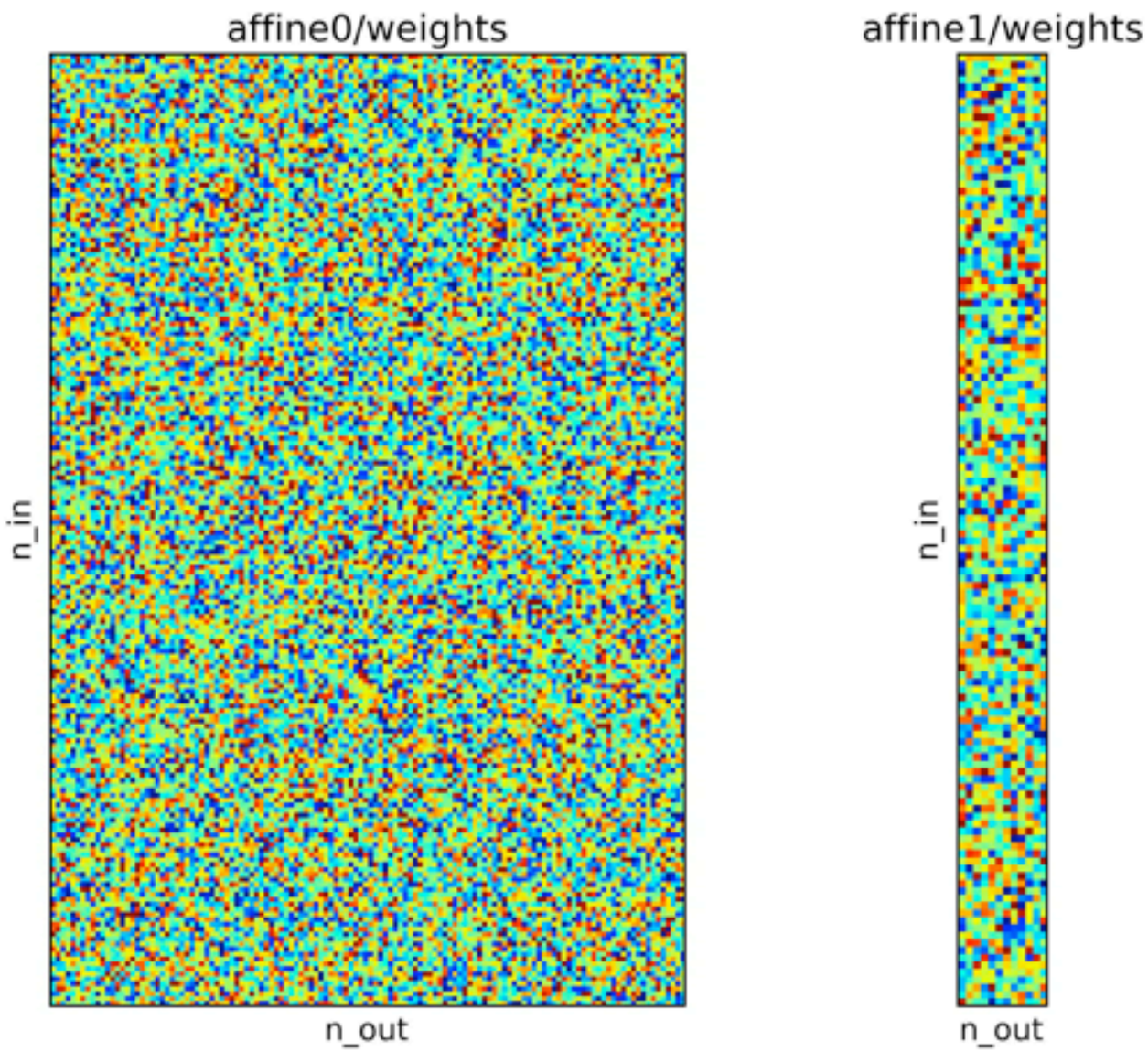
Tricks and tips for training

- Leverage of known data distortions
  - CQT rotations
  - Perceptual codecs
  - Additive noise
- Tuning hyperparameters: sample distributions rather than grid searches
- Be mindful of latent priors in your data

Controlling complexity

- Regularization can help reign in overly complex models
  - Weight decay
  - Sparsity - weights or representations
  - Parameter normalization / limiting
- Dropout, point-noise, data-driven initialization
- Model capacity can be reduced

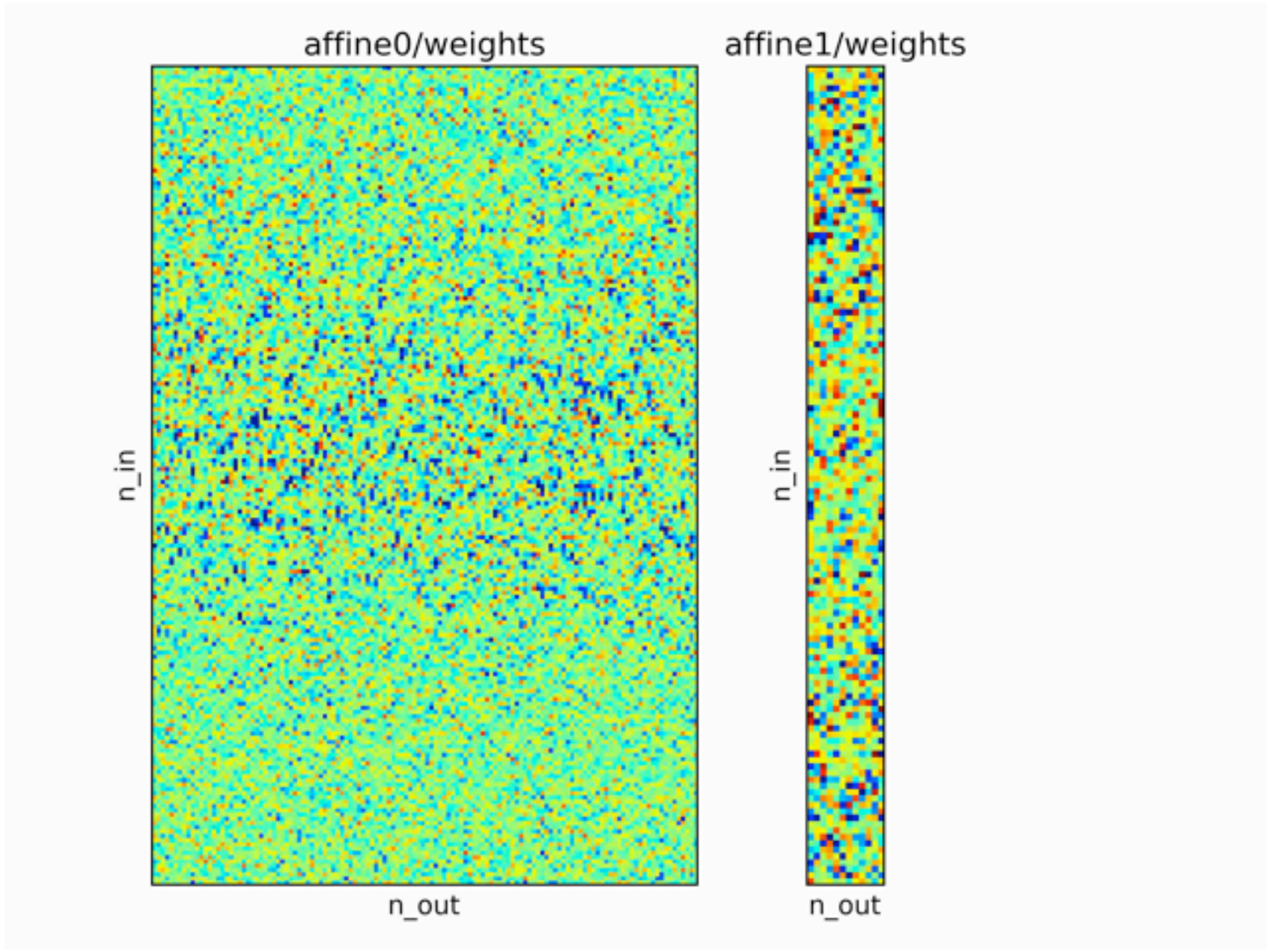
2-Layer Chroma Network - Start



MARL

ismir
Curitiba • Brazil

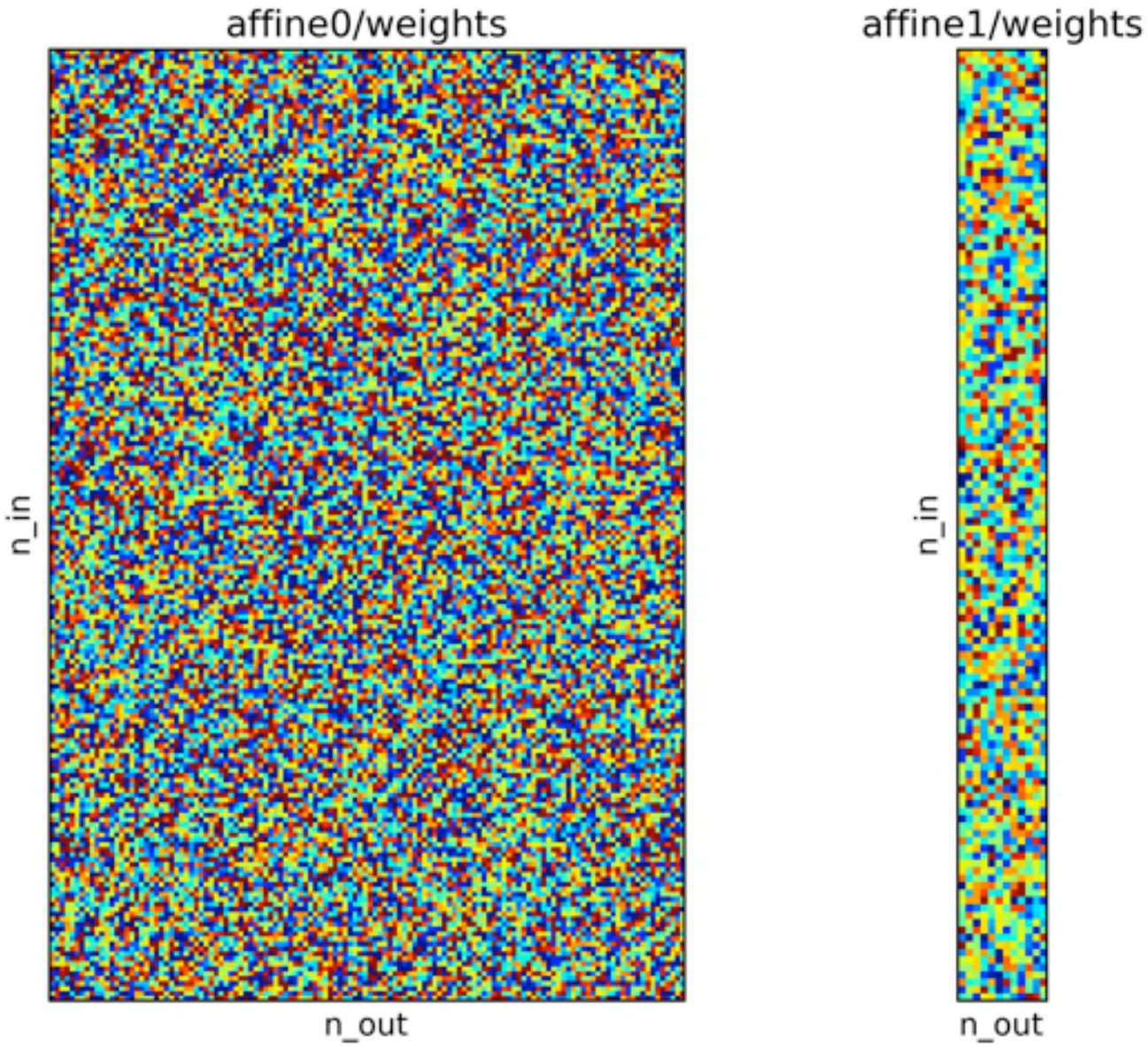
2-Layer Chroma Network - End



MARL

ismir
Curitiba • Brazil

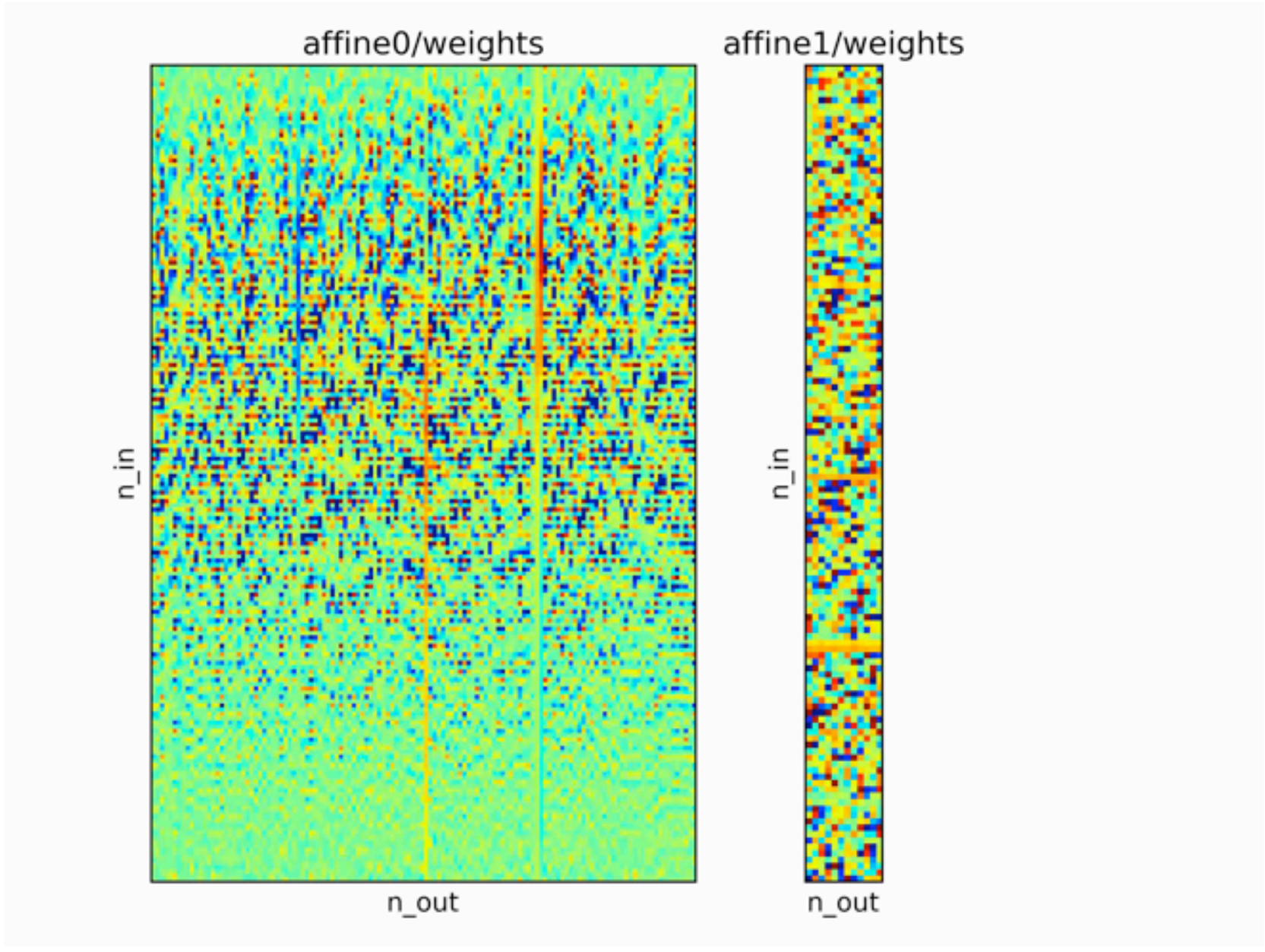
Regularized Learning - Start



MARL

ismir
CURITIBA • BRAZIL

Regularized Learning - End

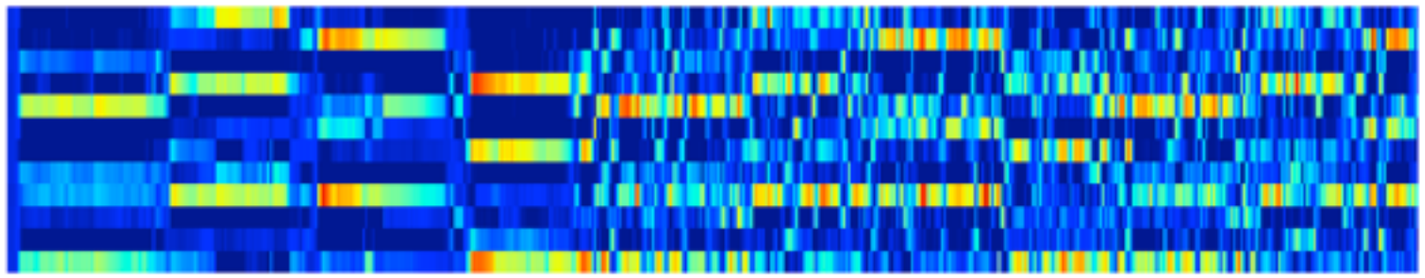


MARL

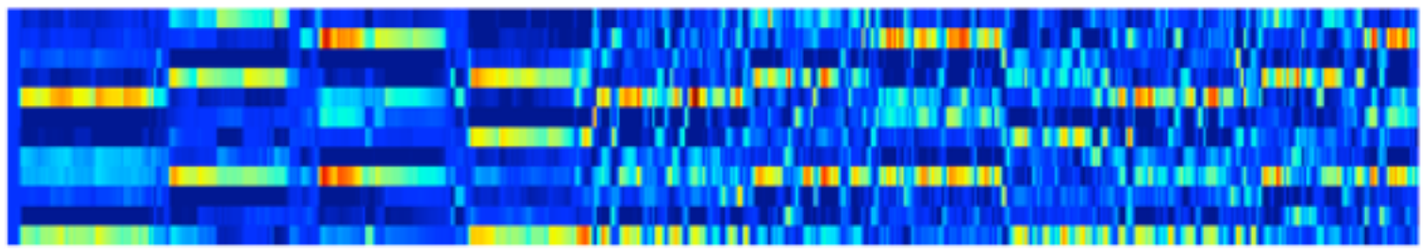
ismir
Curitiba • Brazil

Learned Chroma, side-by-side

Unconstrained



Regularized



MARL

ismir
CURITIBA • BRAZIL

Outline

- In this part of the talk, we'll touch on the following:
  - Recap: What "is" deep learning
  - How you are already doing it (kinda)
  - Why should you consider it
  - Some tips, tricks, insight, and advice
  - How you might do it intentionally
  - A few thoughts for the future

MARL

ismir
Curitiba • Brazil

Slinging code

- Python + Theano make this very easy to try out.
  - Symbolic differentiation
  - Compiles to C-code (wicked fast!)
  - CUDA (GPU) integration
- Stop by the Late-breaking & Demos on Friday for a hack session and install assistance!
  - Chord-annotated DFT dataset
  - Monophonic instrument dataset
  - Sample code for you to use

MARL

ismir
CURITIBA • BRAZIL

```
# ------------------------------------------------------
# Step 1. Build the network
# ------------------------------------------------------
x_input = T.matrix('input')

# Define layer shapes -- (n_in, n_out)
L0_dim = (1025, 256)
L1_dim = (256, 64)
L2_dim = (64, 12)

# Layer 0
weights0 = theano.shared(np.random.normal(scale=0.01, size=L0_dim))
bias0 = theano.shared(np.zeros(L0_dim[1]))
z_out0 = hwr(T.dot(x_input, weights0) + bias0)

# Layer 1
weights1 = theano.shared(np.random.normal(scale=0.01, size=L1_dim))
bias1 = theano.shared(np.zeros(L1_dim[1]))
z_out1 = hwr(T.dot(z_out0, weights1) + bias1)

# Layer 2
weights2 = theano.shared(np.random.normal(scale=0.01, size=L2_dim))
bias2 = theano.shared(np.zeros(L2_dim[1]))
z_output = hwr(T.dot(z_out1, weights2) + bias2)
```

```
# ----------------------------------------------------
# Step 2. Define a loss function
# ----------------------------------------------------
y_target = T.ivector('y_target')
templates = theano.shared(chroma_templates, 'templates')

# Here, we add broadcast dimensions to the output as (batch_size, 1, 12)
#    and the templates as (1, num_templates, 12).
distance = T.pow(
    z_output.dimshuffle(0, 'x', 1) - templates.dimshuffle('x', 0, 1), 2.0)

scalar_loss = T.mean(
    distance[T.arange(y_target.shape[0], dtype='int32'), y_target])
```

```
# ------------------------------------------------------------
# Step 3. Compute Update rules
# ------------------------------------------------------------
eta = T.scalar(name="learning_rate")
updates = OrderedDict()
for param in [weights0, bias0, weights1, bias1, weights2, bias2]:
    # Compute the gradient with respect to each parameter.
    gparam = T.grad(scalar_loss, param)
    # Now, save the update rule for each parameter.
    updates[param] = param - eta * gparam
```

```
# ----------------------------------------------
# Step 4. Compile wicked fast theano functions!
# ----------------------------------------------
update_fx = theano.function(inputs=[x_input, y_target, eta],
outputs=scalar_loss,
updates=updates,
allow_input_downcast=True)
```

And iterate until convergence!

(This has only gotten faster.)

MARL

ismir
Curitiba • Brazil

Outline

- In this part of the talk, we'll touch on the following:
  - Recap: What "is" deep learning
  - How you are already doing it (kinda)
  - Why should you consider it
  - Some tips, tricks, insight, and advice
  - How you might do it intentionally
  - A few thoughts for the future

Tangible Opportunities

- Contribute back to the bigger machine learning community
- Application to problems for which we have little feature insight
  - Timbre
  - Auto-mixing
- Time and sequences are the crux of music
  - Non-linear motion, sequentiality, repetition (long-term structure)
  - Harmonic and temporal correlations
- But we know Digital Signal Processing:
  - Convolutional Networks $\rightleftharpoons$ Normal Filterbanks (FIR)
  - Recurrent Networks $\rightleftharpoons$ Recursive Filterbanks (IIR)

Challenges

- Domain knowledge is crucial to success
  - Can we initialize a network with a state of the art system, e.g. tempo tracking
- Unsupervised learning, i.e. making sense of unlabeled data.
  - Still a good goal! (our brains do it, after all)
  - Reconstruction / computational creativity?
- Music signal processing has advantages over other fields
  - Time is fundamental, but ultimately an open question
  - Strong potential to lay the foundation for better AI
- Analysis of learned functions, insights for music, MIR
- Leverage compositional tools to create music data for training
  - Can we finally solve onset detection, tempo tracking, or multi-pitch estimation?

thanks / questions?

mail: ejhumphrey@nyu.edu
twitter: ejhumphrey

MARL

ismir
CURITIBA • BRAZIL