# Machine learning and signal processing for hyperspectral data classification

Devis Tuia

University of Zürich, Switzerland – devis.tuia@geo.uzh.ch

11.05.2016

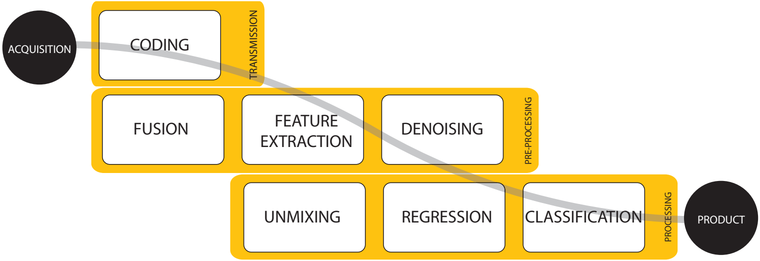
SFPT meeting, Grenoble.

## Menu of the day

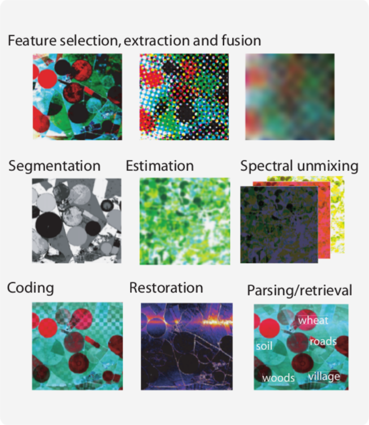
1. **Introduction**
   - The image processing chain
   - Current challenges
2. **Feature extraction from remote sensing images**
   - Spatial feature extraction
   - Spectral feature extraction
3. **Supervised remote sensing image classification**
   - Introduction to supervised image classification
   - Contextual information
   - Multisource image fusionwith LiDAR data
   - Prior knowledge and invariances

## Part 1: Introduction to hyperspectral image processing

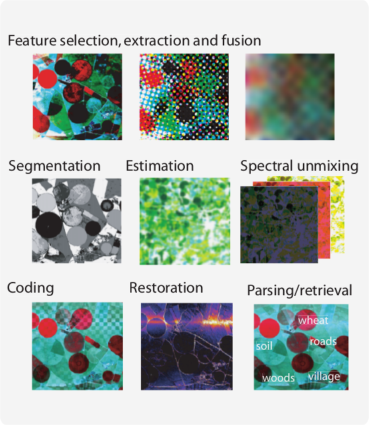
**A standard image processing chain:**



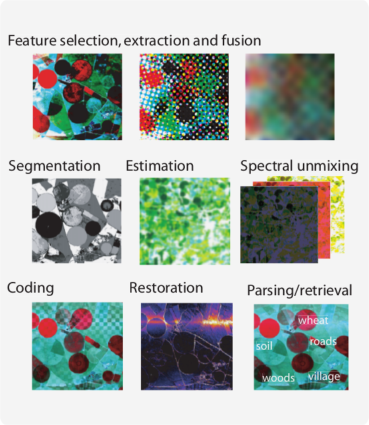

- Many steps and by-products from signal/image acquisition to the product
- Transmission $\longrightarrow$ Preprocessing $\longrightarrow$ Processing
- A wide diversity of problems and dedicated tools



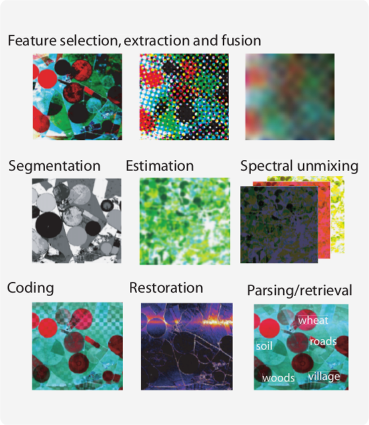

1. **Select best features (channels, spatial) that describe the problem (classification, retrieval)**
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene



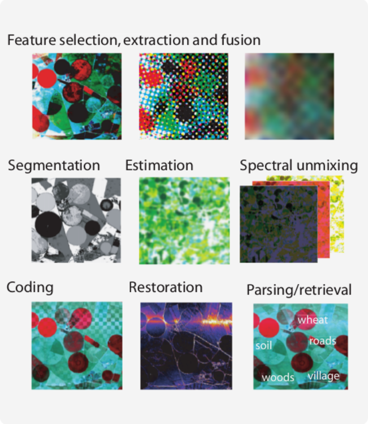

1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene



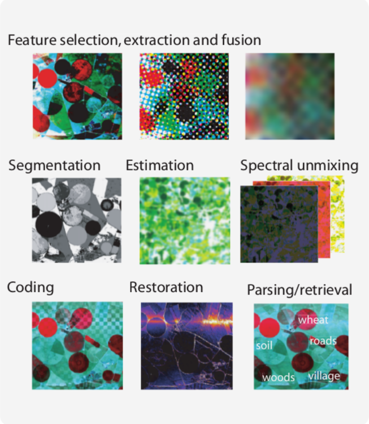

1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene



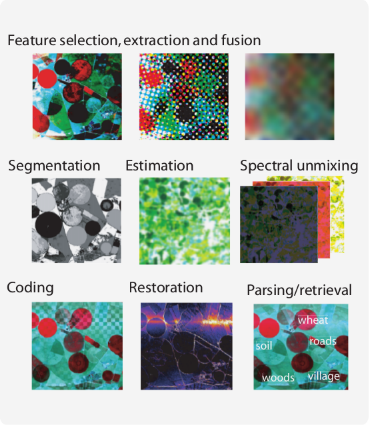

1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene



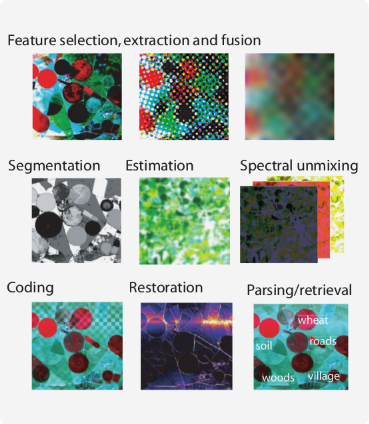

1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene




1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene

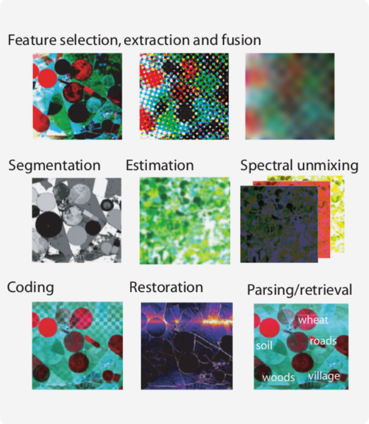



1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene



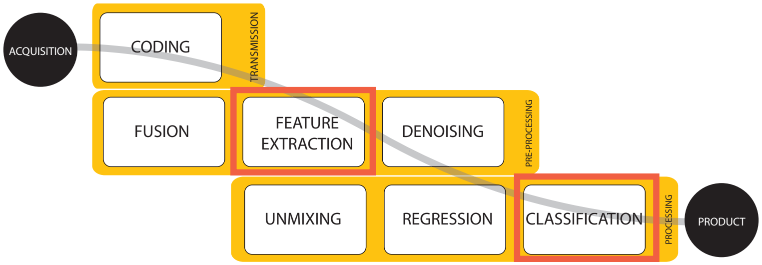

1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene

University of Zurich[UZH]




1. Select best features (channels, spatial) that describe the problem (classification, retrieval)
2. Extract (lin/nonlin) combinations of spectral channels that best describe the problem
3. Combine panchromatic and optical bands to improve products
4. Automatically find groups of pixels in the image (for screening, detection)
5. Estimate geo-bio-physical parameters and variables (temperature, LAI, etc) from spectra
6. Estimate the spectral components (pure pixels, endmembers) in a 'mixed' pixel
7. Compress images for storage and transmission, while keeping most of the information
8. Remove noise and distortions due to acquisition (sun glint) or transmission (vertical stripes)
9. Assign semantic classes to objects (pixels, patches, regions) in the scene

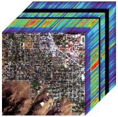

University of Zurich UZH




- Many steps and by-products from signal/image acquisition to the product
- Transmission $\longrightarrow$ Preprocessing $\longrightarrow$ Processing
- A wide diversity of problems and dedicated tools

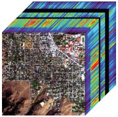
**Characteristics of hyperspectral data:**

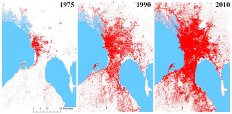
- High spectral resolution → moderate spatial resolutions (mixed pixels, subpixel targets)
- High dimensional data: multi-temporal, multi-angular and multi-source fusion
- Non-linear and non-Gaussian feature relations
- Few supervised (labeled) information is available (high cost)
- Tons of data to process in (near) real-time

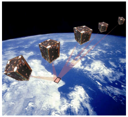


**Characteristics of hyperspectral data:**

- High spectral resolution → moderate spatial resolutions (mixed pixels, subpixel targets)
- High dimensional data: multi-temporal, multi-angular and multi-source fusion
- Non-linear and non-Gaussian feature relations
- Few supervised (labeled) information is available (high cost)
- Tons of data to process in (near) real-time

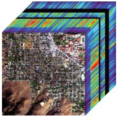


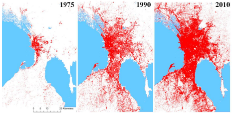


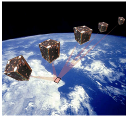



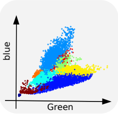
**Characteristics of hyperspectral data:**

- High spectral resolution $\rightarrow$ moderate spatial resolutions (mixed pixels, subpixel targets)
- High dimensional data: multi-temporal, multi-angular and multi-source fusion
- Non-linear and non-Gaussian feature relations
- Few supervised (labeled) information is available (high cost)
- Tons of data to process in (near) real-time





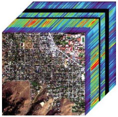


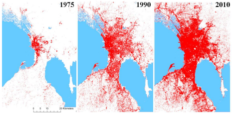


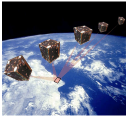

University of Zurich[UZH]

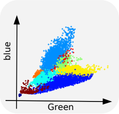
**Characteristics of hyperspectral data:**

- High spectral resolution $\rightarrow$ moderate spatial resolutions (mixed pixels, subpixel targets)
- High dimensional data: multi-temporal, multi-angular and multi-source fusion
- Non-linear and non-Gaussian feature relations
- Few supervised (labeled) information is available (high cost)
- Tons of data to process in (near) real-time



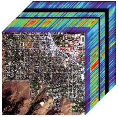


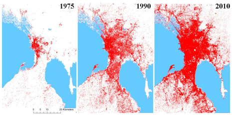



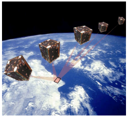


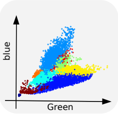

**Characteristics of hyperspectral data:**

- High spectral resolution $\rightarrow$ moderate spatial resolutions (mixed pixels, subpixel targets)
- High dimensional data: multi-temporal, multi-angular and multi-source fusion
- Non-linear and non-Gaussian feature relations
- Few supervised (labeled) information is available (high cost)
- Tons of data to process in (near) real-time

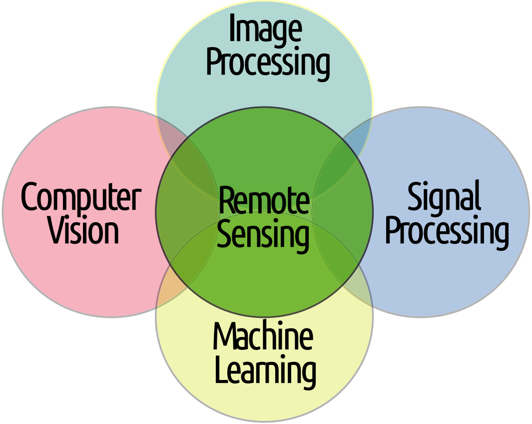







**We will live at the intersection:**

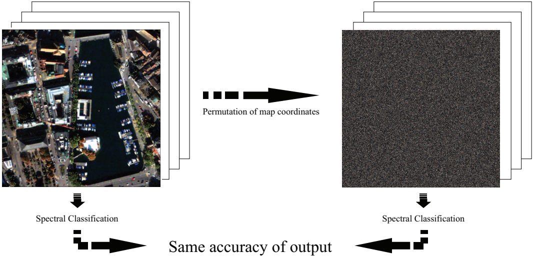


## Part 2: Feature extraction from hyperspectral images

## Why feature extraction?

Extracting features from remote sensing images is essential to:

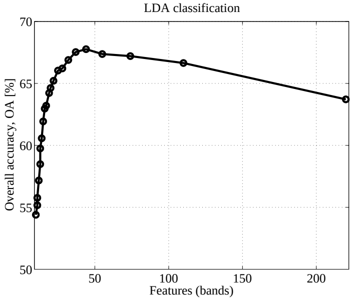
- Compress information for storage/transmission
- Reduce (spatial and spectral) redundancy

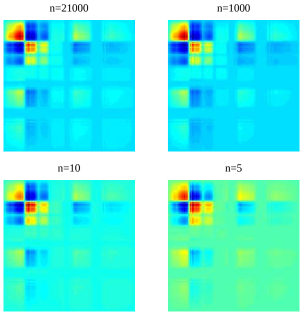



- Make image processing algorithms more robust (to noise, ♯labels vs. dim.)
- Understand the underlying physical relations

## Why feature extraction?

[Hughes69]

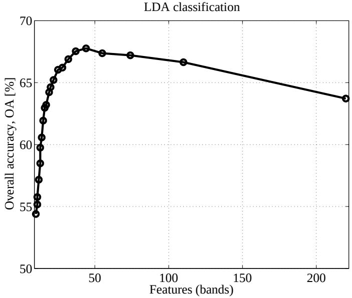


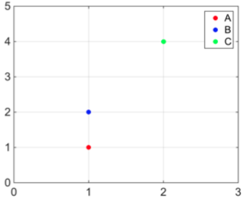



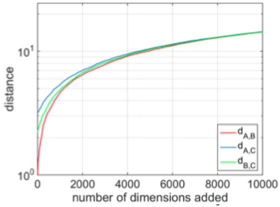

- Algorithms cannot deal with high-dim feature vectors efficiently
- We require fast processing of few richer components
- Many times the spectral information is not enough

## Why feature extraction?

[Hughes69]



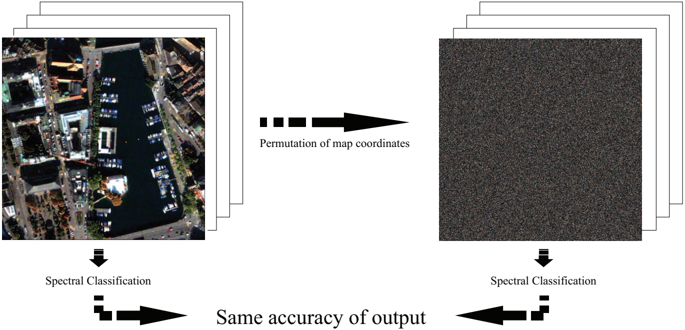

- Algorithms cannot deal with high-dim feature vectors efficiently
- We require fast processing of few richer components
- Many times the spectral information is not enough

## Today we consider:

1. Spatial/contextual
   - Texture
   - Math morphology
2. Spectral: extract features that enforce properties of the data we like
   - Compression: PCA
   - Atmospheric compensation: KEMA

## Why spatio / spectral features

Relying only on spectral information, we disregard the spatial context of the pixels.

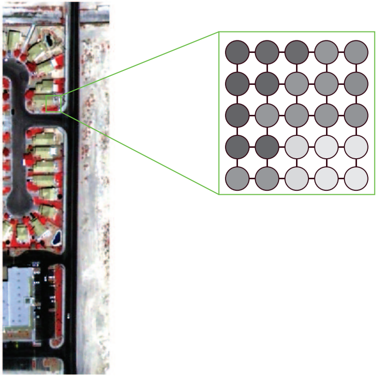


## Going contextual

Let's consider some assumptions

1. Images are intrinsically spatial, not just 'data'
   $>>$ to use the position information makes sense.
2. Objects are sharply separated
   $>>$ contrast can be used to avoid oversmoothing.
3. Classes ($\sim$ objects) tend to be spatially consistent
   $>>$ neighboring pixels tend to belong to the same class.

## 1. Images are spatial *random fields*



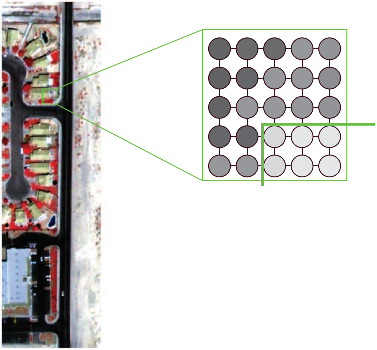
- Gray values vary smoothly in the spatial domain
- They are NOT independent wrt their neighbors

## Going contextual

Let's consider some assumptions

1. Images are intrinsically spatial, not just 'data'
   >> to use the position information makes sense.
2. Objects are sharply separated
   >> contrast can be used to avoid oversmoothing.
3. Classes ($\sim$ objects) tend to be spatially consistent
   >> neighboring pixels tend to belong to the same class.

## 2. Objects are separated by high contrast regions



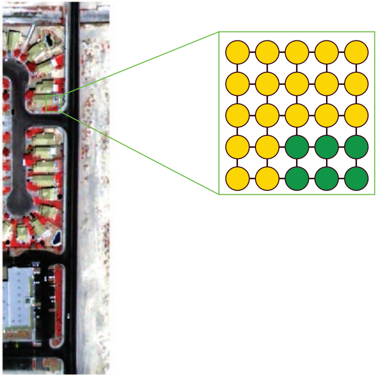
- High gradient is a sharp boundary
- What is beyond is probably another object

## Going contextual

Let's consider some assumptions

1. Images are intrinsically spatial, not just 'data'
   [1.] >> to use the position information makes sense.
2. Objects are sharply separated
   >> contrast can be used to avoid oversmoothing.
3. Classes (∼ objects) tend to be spatially consistent
   >> neighboring pixels tend to belong to the same class.

## 3. Classes are also generally smooth



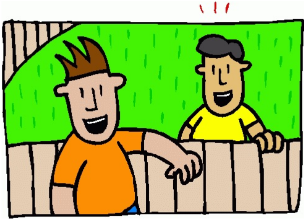
- Neighboring pixels tend to share the same class
- Size and type of the relation depend on many factors (resolution, type of class, ...) > prior information

## Going contextual

Summing up:

1. Images are intrinsically spatial, not just ‘data’
2. Objects are sharply separated
3. Classes ($\sim$ objects) tend to be spatially consistent

It is time to meet the pixels’ neighbors!

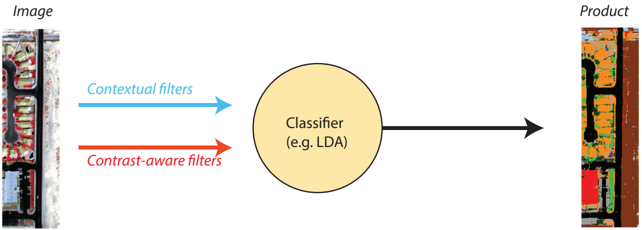




University of Zurich[UZH]

## We can act at the feature level

- We generate relevant filters
- (opt.) We select the good ones
- We classify



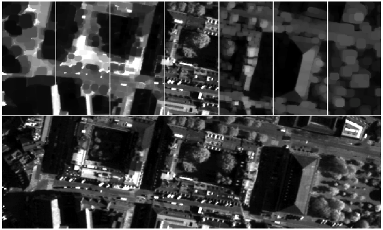



University of Zurich[UZH]

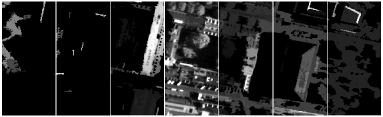
## Filters

- Signal modifications that smooth or enhance edges.

Morphological opening and closing



Morphological reconstruction

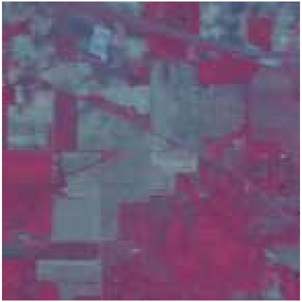


Attribute filtering

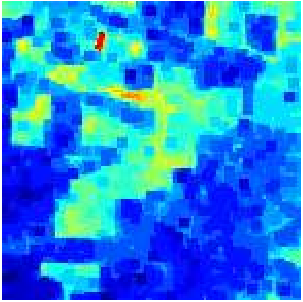
- Physics-inspired indices: NDVI, Red-edge, NDWI, ...

**Erosion:** “Replace pixel with the minimum surrounding pixel over SE.”

```
>> se = strel('disk',3); O = imerode(I,se);
```



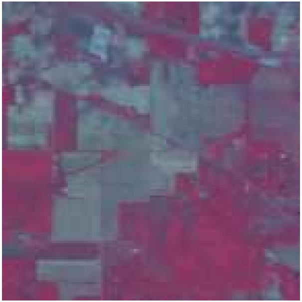
Erosion, disk 3x3



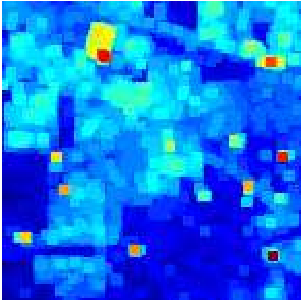
- Darker features than the surroundings are enlarged
- Brighter features than the surroundings shrink

**Dilation:** "Replace pixel with the maximum surrounding pixel over SE."

```
>> se = strel('disk',3); O = imdilate(I,se);
```



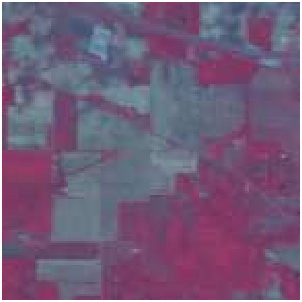
Dilation, disk 3x3



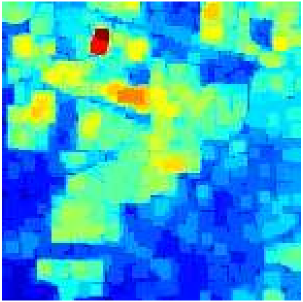
- Brighter features than the surroundings are enlarged
- Darker features than the surroundings shrink

**Opening:** "Erosion followed by dilation"

```
>> se = strel('disk',3); O = imopen(I,se);
```



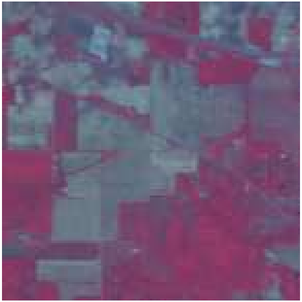
Opening, disk 3x3



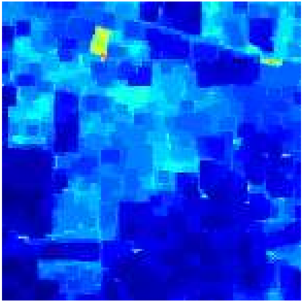
- Brighter features than the surroundings and smaller than the SE disappear
- Other features (dark, or bright and large) remain unchanged

**Closing:** "Dilation followed by erosion."

```
>> se = strel('disk',3); C = imclose(I,se);
```



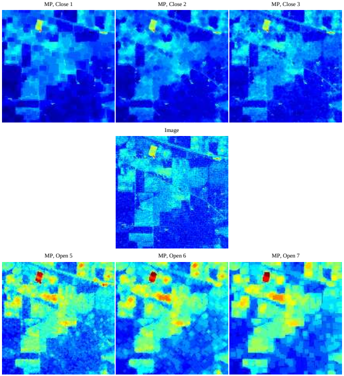
Closing, disk 3x3



- Darker features than the surroundings and smaller than the SE disappear
- Other features (bright, or dark and large) remain unchanged

**Morphological profile:** "Openings and closings with increasing SE"

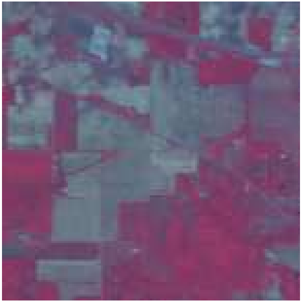
```
>> se = strel('diamond',5); repeat opening-closing operations;
```



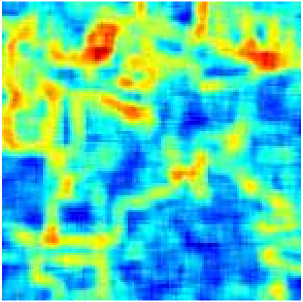

- Pixels turn into a sequential analysis of fine-to-coarse relations
- Useful as a feature vector for processing (e.g. classification)

**Local entropy:** "Replace a pixel with the entropy value of the neighborhood"

```
>> H = entropyfilt(I/max(I(:)));
```



Local entropy, 9x9 window



- Useful for edge detection
- Useful for saliency and detection of anomalies



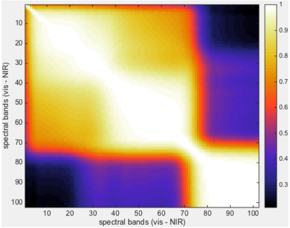
University of Zurich[UZH]

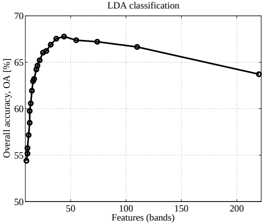
## Spectral feature extraction

Ok, now we know how to extract filters. But on which bands?

- In hyperspectral images we have hundreds to thousands of features!
- Extracting filters for each would lead to millions of **redundant** features (the bands are collinear)

↑




↓

- And classifiers will be less accurate (Hughes phenomenon)
- And slower...

... A solution can be found in ...

## Dimensionality reduction (a.k.a **feature extraction**)

- We want to recombine information of the image into some features that show some properties of interest for us
- Most of the spectral feature extractors are based on multivariate analysis: *"project data onto a subspace that maximizes explained variance, minimize classification error, etc."*

Today we consider two problems:

1. Compressing the information: principal component analysis (PCA)
2. Making classes more similar: kernel manifold alignment (KEMA)

## Principal component analysis (PCA)

- *"Find projections maximizing the variance of the data:"*

$$\text{PCA:} \quad \begin{array}{ll} \text{maximize:} & \mathrm{Tr}\{(\mathbf{XU})^\top(\mathbf{XU})\} = \mathrm{Tr}\{\mathbf{U}^\top \mathbb{C}_{xx}\mathbf{U}\} \\ \text{subject to:} & \mathbf{U}^\top\mathbf{U} = \mathbf{I} \end{array}$$

- **The Matlab PCA code:**

```
>> C = cov(X);
>> [U L] = eigs(C,d);
>> Xtest_projected = Xtest*U;
>> Xtest_projected = Xtest*U(:,1:np);
```

- **Pros & cons:**
  - √ Simplicity
  - √ Easy to understand
  - √ Leads to convex optimization problems
  - × Unsuitable for non-linear problems
  - × More dimensions than points?

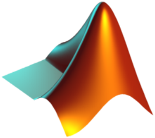
## Principal component analysis (PCA)

- *"Find projections maximizing the variance of the data:"*

$$\text{PCA:} \qquad \text{maximize:} \quad \mathrm{Tr}\{(\mathbf{XU})^\top(\mathbf{XU})\} = \mathrm{Tr}\{\mathbf{U}^\top \mathbb{C}_{xx}\mathbf{U}\}$$
$$\text{subject to:} \quad \mathbf{U}^\top\mathbf{U} = \mathbf{I}$$

- **The Matlab PCA code:**

```
>> C = cov(X);
>> [U L] = eigs(C,d);
>> Xtest_projected = Xtest*U;
>> Xtest_projected = Xtest*U(:,1:np);
```

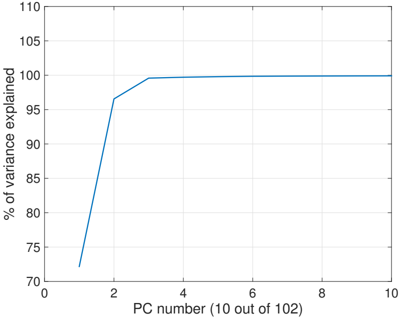


- **Pros & cons:**
  - √ Simplicity
  - √ Easy to understand
  - √ Leads to convex optimization problems
  - × Unsuitable for non-linear problems
  - × More dimensions than points?



University of Zurich[UZH]

## An Example: Pavia data (with video!)

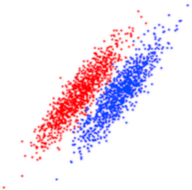





University of Zurich[UZH]

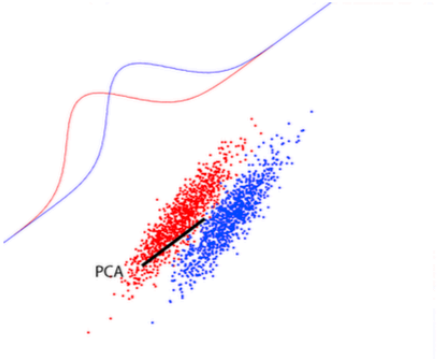
## Using feature extraction for other objectives

- PCA compacts information
- It is useful if you want to reduce the dimensionality and have informative features to extract spatial indices
- It has **nothing** to do with classification: the features are not discriminative



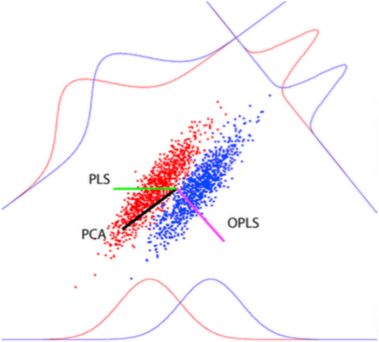
## Using feature extraction for other objectives

- PCA compacts information
- It is useful if you want to reduce the dimensionality and have informative features to extract spatial indices
- It has **nothing** to do with classification: the features are not discriminative



## Using feature extraction for other objectives

- PCA compacts information
- It is useful if you want to reduce the dimensionality and have informative features to extract spatial indices
- It has **nothing** to do with classification: the features are not discriminative Other feature extractors are discriminative: PLS, OPLS



## Orthonormalized PLS (OPLS)

- *"OPLS chooses the projection* $\mathbf{U}$ *that minimizes the MSE error using a linear regression:"*

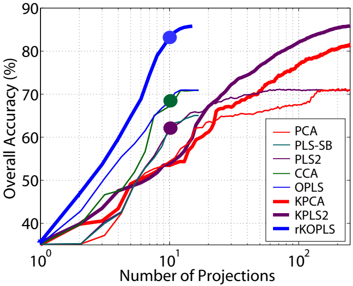
$$\text{OPLS:} \qquad \begin{array}{ll} \text{find:} & \mathbf{U} = \arg\,\min\{\|\mathbf{Y} - (\mathbf{XU})\mathbf{W}\|_F^2\} \\ \text{where:} & \mathbf{W} = (\mathbf{XU})^\dagger \mathbf{Y} = ((\mathbf{XU})^\top \mathbf{XU})^{-1} \mathbf{XUY} \end{array}$$

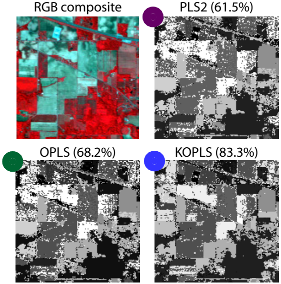
- **The Matlab OPLS code**

```
>> [U,D] = eig((X'*Y)*(Y'*X),X'*X);
>> Xtest_projected = Xtest*U;
>> Xtest_projected = Xtest*U(:,1:np);
```

## Experimental comparison

- **Data:**
  - AVIRIS image taken over NW Indiana's Indian Pine test site in June 1992
  - $145 \times 145$ image size, 220 features (bands), 16 land cover classes
  - 80% for training and 20% for testing
  - Classifier: linear classifier on top of different number of features
- **Results:**




  - Supervised feature extraction often better than unsupervised
  - Higher accuracies lead to smoother maps
  - kOPLS excels in performance, needs few components
  - kOPLS reduce false alarm rates in large homogeneous vegetation areas


University of Zurich[UZH]

## Using feature extraction for other objectives: shadow compensation

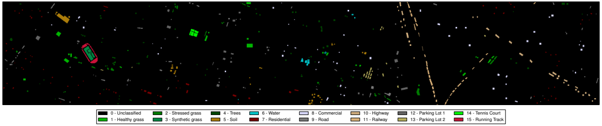
What if our data show undesired spectral effects?

- PCA compacts information
- PLS / OPLS / ... provide discriminative bands
- can we define projections that perform automatic relative normalization?
- like an histogram matching between images, or an automatic atmospheric correction

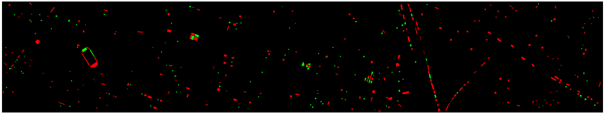
## Using feature extraction for other objectives: shadow compensation

- We have one hyperspectral image form CASI, a part has a strong shadow

- Our ground reference :




- The train ground (↓ green) truth is only on lit pixels
- The test ground truth (↓ red) is a mixture of lit pixels and under shadow



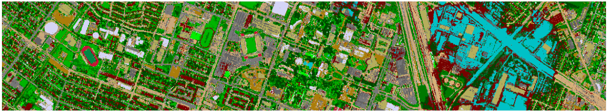


University of Zurich[UZH]

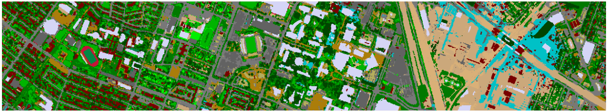
## Using feature extraction for other objectives: shadow compensation

- We have one hyperspectral image form CASI

- A classifier will do something like that (OA: 71%, 4% under the shadow):



- If we add LiDAR and spatial filter (OA: 85%, 23% under the shadow):



## Kernel Manifold Alignment

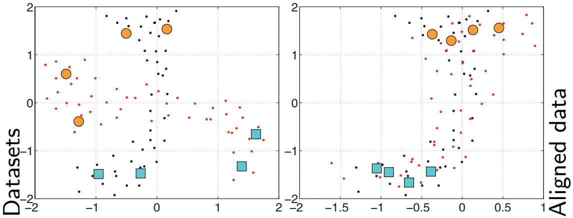
The Kernel Manifold Alignment registers images spectrally. It searches for projections that

- A: Maintain the original spectral neighoborhood relationships (keep the reflectance structures)
- B: Pull samples of the same class close
- C: Push samples of different classes apart

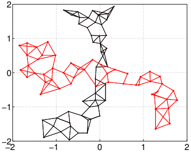
Ref. Tuia and Camps Valls: Kernel manifold alignment for domain adaptation. PLoS One, 2016.

CODE: `https://github.com/dtuia/KEMA`

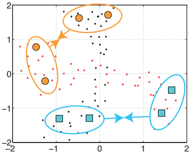
## Kernel Manifold Alignment (intuition)



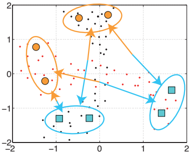

Under constraints:



Same geometry



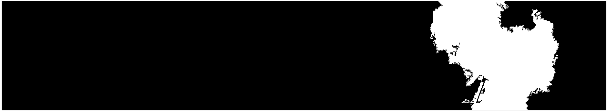
Pull classes close



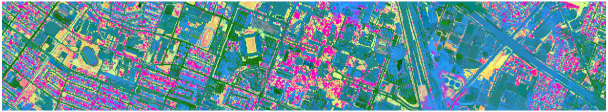
Push classes away

## Using feature extraction for other objectives: shadow compensation

- We register **spectrally** the illuminated and shadowed parts



- This is how the three first projections look like



- A classifier after projection (OA: 83.8%, 70% under the shadow):

- If we add LiDAR and spatial filter (OA: 94.3%, 91% under the shadow):



University of Zurich[UZH]

## Feature extraction: summary

- Extracting features from remote sensing images is essential to:
  - Compress information for storage/transmission
  - Reduce (spatial and spectral) redundancy
  - Visualize data characteristics
- Spectral features rely either on physical prior knowledge or statistical techniques that optimize a sensible criterion
- Spatial features rely on image processing operations building on the classical smoothness assumption in the image space or detect edges
- All in all, they always make the problem better posed, so use them!

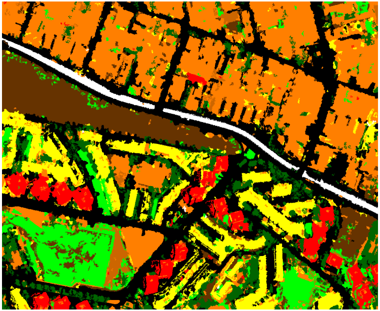
## Part 3: Supervised hyperspectral image classification

## In a nutshell

From here

Gray values, $f(light)$
$\mathbb{R}^b$

To here



Limited number of classes
$\mathbb{Z}$

[Tuia et al, Classification of very high spatial resolution imagery using mathematical morphology and support vector machines, IEEE TGRS, 2009]

## Did you say classification?

Need for generalization of images for:

- land cover / coastal monitoring
- post catastrophe assessment
- military applications
- population movements, urban growth, policy making

Need for automatic routines because

- the human brain is excellent at pattern recognition
- for a computer,
  - a pixel is just a stack of values (one per feature)
  - the notion of object does not exist a priori.

## Statistical classifiers have been readily applied to the problem:

**Parametric**

Assume a particular density distribution
LDA, GMM

**Non-parametric**

No assumption about the data distribution
*k*-NN, NNETS, TREES, SVM

**Supervised**

Need labeled input-output pairs
LDA, *k*-NN, TREES, SVM

**Unsupervised**

No need labels
*k*-means, EM-GMM, SOM

**Semisupervised**

Use both labeled and unlabeled data
Laplacian SVM, TSVM, graphs

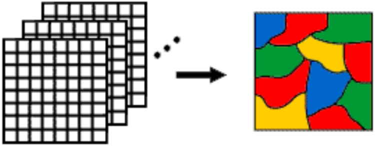
**One-class**

Interest in detecting just one class
SAM, OSP, RX, OC-SVM

- Not too much success in parametric classifiers, as some assumptions break
- Currently, nonparametric classifiers and committees of experts excel!
- *k*-NN: good compromise between accuracy and computational cost
- Support vector machines (SVM) typically outperform the rest

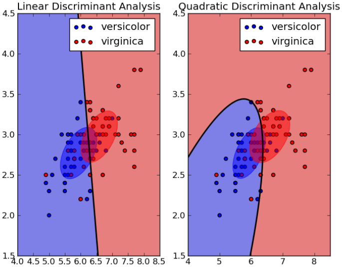
## Classifiers:

- Linear discriminant analysis (linear, quadratic, Mahalanobis)
- $k$-Nearest neighbors (KNN)
- Random Forests (RF)
- Support Vector Machines (SVM)



## Linear discriminant analysis (LDA): "Fits a Gaussian to each class data"

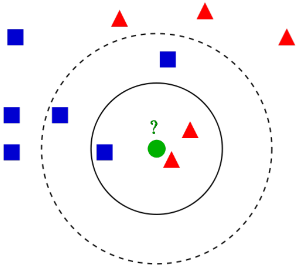
- Linear discriminant analysis ('linear'): Fit a multivariate Gaussian to each group/class through a joint covariance matrix
```
>> yp=classify(Xtest,Xtrain,Ytrain,'linear');
```
- Linear discriminant analysis ('quadratic'): Fit a multivariate Gaussian to each group/class through a class-dependent covariance matrix
```
>> yp=classify(Xtest,Xtrain,Ytrain,'quadratic');
```



## $k$ nearest neighbor ($k$-NN)

- non-parametric memory-based (lazy) classifier
- assigns the test label from the closest training point(s)
- we can play around with the notion of distance (e.g. Euclidean, SAM, etc.)
- $k$-NN is a rather slow method with many samples and high $k$
- $k = 1$ use to work in real applications!

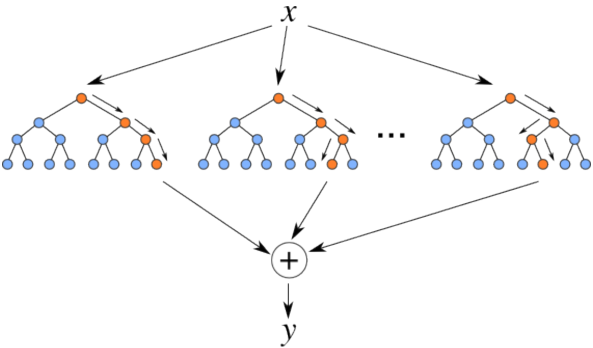
```
>> mdl = fitcknn(Xtrain,Ytrain,'NumNeighbors',1);
>> yp = predict(mdl,Xtest);
```



## Random Forests (RF)

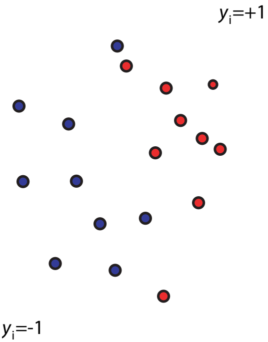
- Trains a set of `Ntrees` decision trees built on subsets of data and features
- Final prediction is a vote over the trees responses
- More trees is better (more independence), but also slower.
- More depth of the trees tends to overfit.

```
>> RF = TreeBagger(NTrees,Xtrain,Ytrain);
>> [yp,scores] =predict(RF,Xtest);
```

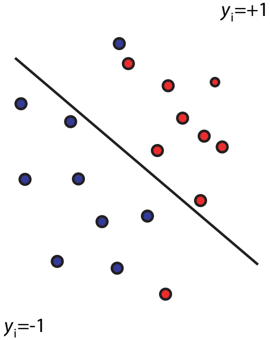


**Support Vector Machines (SVM): "non-parametric kernel method that fits an optimal linear hyperplane separating the classes in a higher dimensional representation (feature) space"**

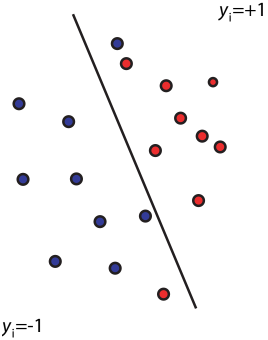
- **Data:** Given $n$ examples $\mathbf{x}_i \in \mathbb{R}^B$ and $y_i \in \{-1, +1\}$ (classes)
- **Objective:** Linear classifier in Hilbert space, $\hat{y} = sign(\mathbf{w}^\top \phi(\mathbf{x}) + b)$.



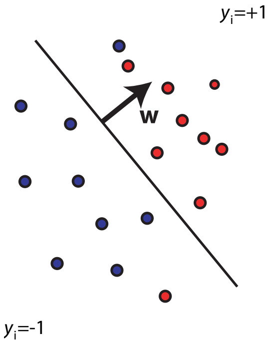
- **Several solutions exist!**
- **Objective:** Define the optimal one $(\mathbf{w}, b)$



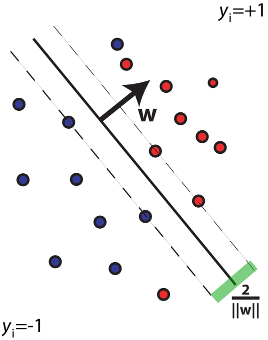
- **Several solutions exist!**
- **Objective:** Define the optimal one $(\mathbf{w}, b)$



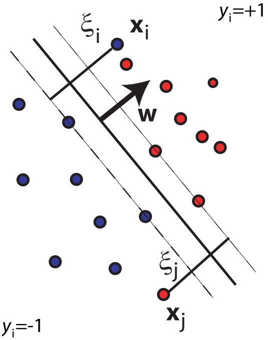
- **Intuitively there's an optimal one!**
- **Objective:** Define the optimal one ($\mathbf{w}$, $b$)



- **Maximize margin separation = minimize $\|\mathbf{w}\|$:** $\min_{\mathbf{w}} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 \right\}$

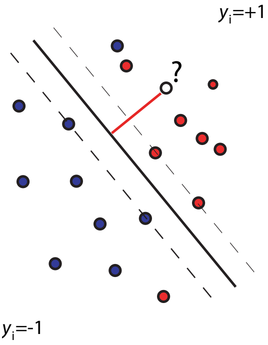


- **Errors must be also penalized!** $\min_{\mathbf{w}} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i \right\}$



$$\hat{y}_j = f(\mathbf{x}_j) = sign(\mathbf{w}^\top \phi(\mathbf{x}_j) + b) = sign\left(\sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{x}_j, \mathbf{x}_i \rangle + b\right)$$

- Instead of computing the exact position of the point w.r.t the hyperplane
- We compute it relatively to the support vectors
- **Support vectors:** the samples on the margin

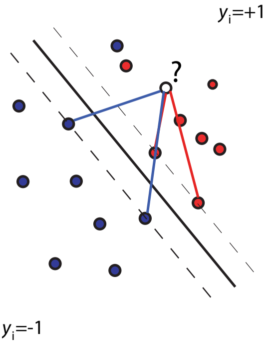




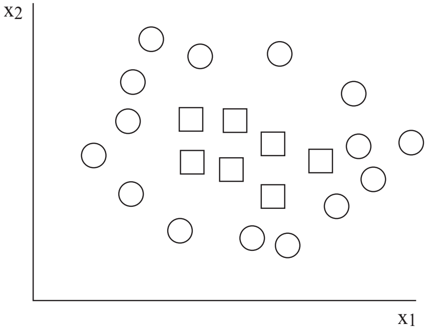
University of Zurich[UZH]

$$\hat{y}_j = f(\mathbf{x}_j) = sign(\mathbf{w}^\top \phi(\mathbf{x}_j) + b) = sign\left(\sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{x}_j, \mathbf{x}_i \rangle + b\right)$$

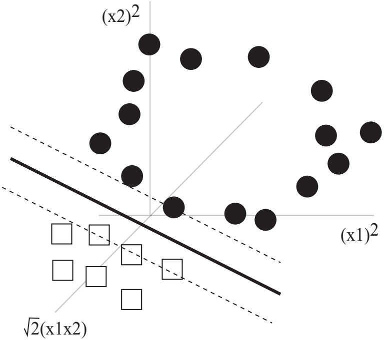
- **Instead of computing hte exact position of the point w.r.t the hyperplane**
- We compute it relatively to the support vectors
- **Support vectors:** the samples on the margin



## But this is only linear. How to solve this?



## 2 possibilities

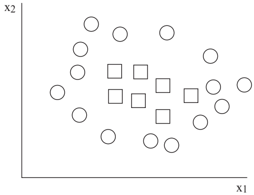


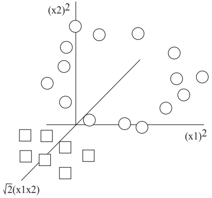

Build a nonlinear model (e.g. NN)

Ask an old friend

**Original space $\mathcal{X}$**

**Feature space $\mathcal{H}$**



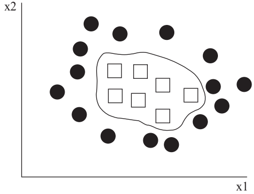



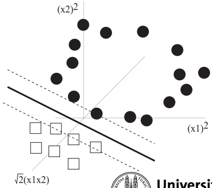

$\rightarrow$ Project $\rightarrow$

$\downarrow$

Use linear model

$\downarrow$




$\leftarrow$ Back in $\mathcal{X}$ $\leftarrow$

## But how?

- The expression of the projecting function $\phi(\mathbf{x})$ can be complicated
- There are infinitely many possible candidates

- Trick: use samples in the original space, get the projected solution

## By using kernels

- A kernel between two samples correspond to their similarity in a higher dimensional space

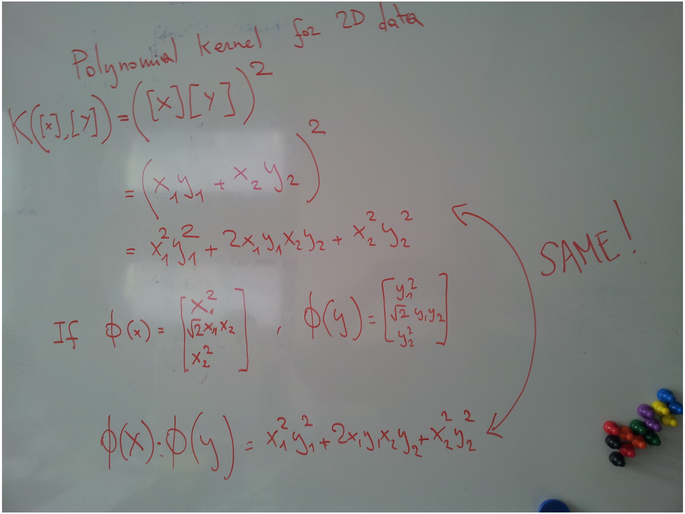
$$K : \mathbf{x} \rightarrow \phi(\mathbf{x})$$

- We evaluate the function on the input samples, and we get their similarity in the projected one

$$K(\mathbf{x}_1, \mathbf{x}_2) = \langle \phi(\mathbf{x}_1), \phi(\mathbf{x}_2) \rangle$$

- e.g squared polynomial kernel in 2D corresponds to a projection on a 3D space

$$\mathbf{x} \in \mathbb{R}^2 = [x_1, x_2]$$
$$\phi(\mathbf{x})_{poly,2} \in \mathbb{R}^3 = [(x_1)^2, \sqrt{x_1 x_2}, (x_2)^2]$$


Polynomial kernel for 2D data
$K([x],[y]) = ([x][y])^2$
$= (x_1 y_1 + x_2 y_2)^2$
$= x_1^2 y_1^2 + 2 x_1 y_1 x_2 y_2 + x_2^2 y_2^2$
SAME!
If $\phi(x) = \begin{bmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{bmatrix}$, $\phi(y) = \begin{bmatrix} y_1^2 \\ \sqrt{2} y_1 y_2 \\ y_2^2 \end{bmatrix}$
$\phi(x) \cdot \phi(y) = x_1^2 y_1^2 + 2 x_1 y_1 x_2 y_2 + x_2^2 y_2^2$

**Support Vector Machines (SVM): “non-parametric kernel method that fits an optimal linear hyperplane separating the classes in a higher dimensional representation (feature) space”**

$$\hat{y}_j = f(\mathbf{x}_j) = sign(\mathbf{w}^\top \phi(\mathbf{x}_j) + b) = sign\left( \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_j, \mathbf{x}_i) + b \right)$$

- **The solution is sparse:** only few examples $\mathbf{x}_i$ with $\alpha_i \neq 0$ are important
- **Support vectors:** define the margin and are misclassified examples
- **The solution is linear in the projected space, but nonlinear in the original one**

## Example: Spatial-spectral multispectral image classification

- Multispectral image: 9 crop classes, Zürich, 2002.
- Quickbird sensor: 4 bands + 22 spatial features (top/bottom hat).
- *Both spatial and spectral information is considered.*

**Accuracy and robustness with contextual information:**

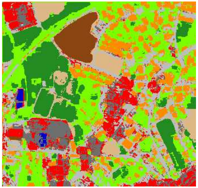
| Training pixels | | OA [%] | | | | | Kappa | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | LDA | 1 Tree | $k$-NN | SVM | NN | LDA | 1 Tree | $k$-NN | SVM | NN |
| 115 | $\mu$ | 72.93 | 71.00 | 75.69 | **83.37** | 77.37 | 0.67 | 0.65 | 0.70 | **0.80** | 0.72 |
| | $\sigma$ | (2.85) | (2.97) | (1.28) | (2.40) | (2.48) | (0.03) | (0.03) | (0.02) | (0.03) | (0.03) |
| 255 | $\mu$ | 77.23 | 73.47 | 80.53 | **85.91** | 80.61 | 0.72 | 0.68 | 0.76 | **0.83** | 0.76 |
| | $\sigma$ | (1.41) | (1.64) | (1.34) | (1.94) | (0.99) | (0.02) | (0.02) | (0.02) | (0.02) | (0.01) |
| 1155 | $\mu$ | 78.35 | 80.45 | 87.32 | **88.03** | 84.29 | 0.74 | 0.76 | 0.84 | **0.85** | 0.81 |
| | $\sigma$ | (0.69) | (0.73) | (0.63) | (1.68) | (1.77) | (0.01) | (0.01) | (0.01) | (0.02) | (0.02) |
| 2568 | $\mu$ | 78.61 | 81.59 | **87.26** | 87.17 | 85.10 | 0.74 | 0.77 | **0.84** | **0.84** | 0.82 |
| | $\sigma$ | (0.57) | (0.89) | (0.61) | (0.85) | (1.05) | (0.01) | (0.01) | (0.01) | (0.01) | (0.01) |



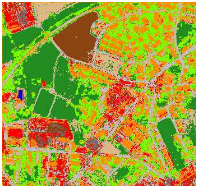
University of Zurich[UZH]

Ground truth

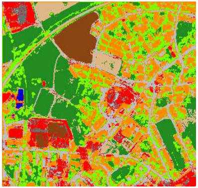
LDA (78.35, 0.74)



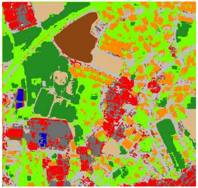
Decision tree (80.45, 0.76)



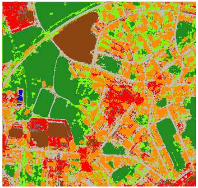
*k*-NN (87.32, 0.84)



**SVM (88.03, 0.85)**



Neural net (84.29, 0.81)



- SVM and *k*-NN detect all major structures of the image
- McNemar's test confirmed visual estimation of the quality

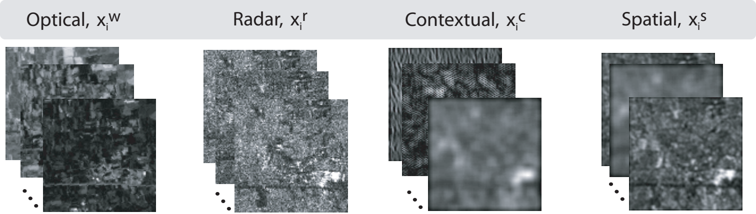
[Camps-Valls et al., Remote Sensing Image Processing, Morgan and Claypool, 2011]

## Image classification needs strong regularization:

- SVM imposes regularization naturally by maximum margin
- RF impose regularization naturally by the ensemble of weak learners
- Advanced classification focuses on other forms of regularization:
  - Reduce dimensionality via feature selection and extraction [+Before+]
  - Include synthetically generated data encodes invariance properties [+Next+]
  - Impose spatial homogeneity of images: include spatial information [+Next+]
  - Include information contained in unlabeled samples
  - Include multisource data: SAR, LiDAR [+Next+]
  - Include ancillary information from expert's knowledge (VIs, ecosystems maps, climate regions, etc)

## How to integrate multi-source information?

- Spatial features
- Textural features
- Time-varying features
- Multi-sensor features
- Multi-angular features

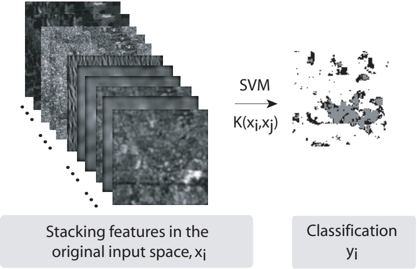


## Stacked approach

- Stacking features that characterize a pixel:

$$\mathbf{x}_i \leftarrow [\mathbf{x}_i^{\omega}, \mathbf{x}_i^{c}, \mathbf{x}_i^{r}, \mathbf{x}_i^{\rho}, \mathbf{x}_i^{s}, \mathbf{x}_i^{t}, \ldots]$$

- Compute matrix $K$ and solve an SVM with the new samples $\mathbf{x}_i$.




- **Problems:**
  1. Dimensionality of the samples is increased extraordinarily!
  2. Cross-relationships among features are not taken into account.
  3. This would be impractical for neural networks, for example.

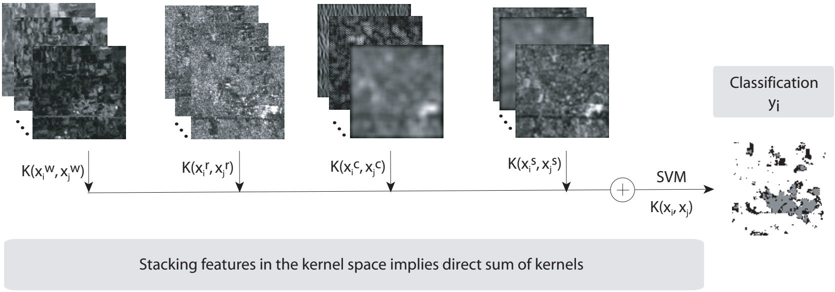


University of Zurich[UZH]

## Kernel-based spatial-spectral HSI classification

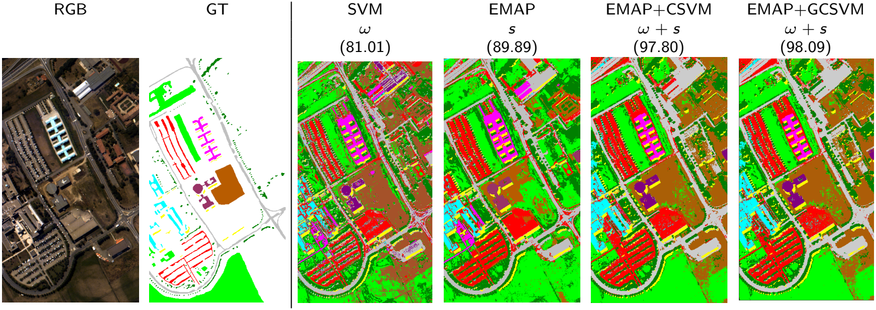
- Some properties of kernel methods (and SVM):

$$K(\mathbf{x}_i, \mathbf{x}_j) = K_1(\mathbf{x}_i, \mathbf{x}_j) + K_2(\mathbf{x}_i, \mathbf{x}_j)$$
$$K(\mathbf{x}_i, \mathbf{x}_j) = K_1(\mathbf{x}_i, \mathbf{x}_j) \cdot K_2(\mathbf{x}_i, \mathbf{x}_j)$$
$$K(\mathbf{x}_i, \mathbf{x}_j) = \eta K_1(\mathbf{x}_i, \mathbf{x}_j), \quad \eta > 0$$




[Tuia et al., Learning relevant image features with multiple kernels classification, IEEE TGRS, 2010]

## Combine advanced spatial features and composite SVM




- ROSIS-03 Pavia University area data set (103 spectral channels and spatial resolution 1.3m), 9 classes
- Spatial components:

| | |
|---|---|
| Benediktson11 | Extended Morphological Profiles (EMAP) |
| CampsValls06 | Cross-kernels composite SVM (CSVM) |
| Li13 | Generalized composite kernels (GCSVM) |

[Li et al., Generalized Composite Kernel Framework for Hyperspectral Image Classification, TGRS 2013]

## GRSS DF-TC competition 2013:

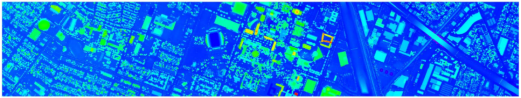
- HSI from CASI1500 sensor (144 bands, 380–1050 nm)
- LiDAR-derived digital surface model (DSM), spatial res. 2.5 m
- 15 classes, challenging problem, diversity of classes
- DSM represents elevation (in [m]) above sea level (Geoid 2012 A model)
- Note a large cloud shadow, validation samples are also there!

### Classes

| Class name | Training set | Test set | Class color |
|---|---|---|---|
| Healthy grass | 198 | 1053 | |
| Stressed grass | 190 | 1064 | |
| Synthetic grass | 192 | 505 | |
| Tree | 188 | 1056 | |
| Soil | 186 | 1056 | |
| Water | 182 | 143 | |
| Residential | 196 | 1072 | |
| Commercial | 191 | 1053 | |
| Road | 193 | 1059 | |
| Highway | 191 | 1036 | |
| Railway | 181 | 1054 | |
| Parking lot 1 | 192 | 1041 | |
| Parking lot 2 | 184 | 285 | |
| Tennis court | 181 | 247 | |
| Running track | 187 | 473 | |

### HSI + LiDAR-derived DSM

(a)



(b)

Credits: Figures from Debes, et al. IEEE-JSTARS 2013. Special thanks to Dr. Saurabh Prasad @ University of Houston, USA.

Data freely available on `http://hyperspectral.ee.uh.edu/?page_id=459`

## Setup

- Training on 2'832 samples
- Testing on spatially separated 12'197 samples
- Classifier: SVM, RBF kernel

- Max of 276 features:
  - 144 spectral bands (shadow corrected)
  - 64 morphological filters from HSI (1st PCA)
  - 1 DSM from LiDAR
  - 64 morphological filters from LiDAR
  - 3 variations of NDVI index
- 8 experiments, acocunting for the different features sets

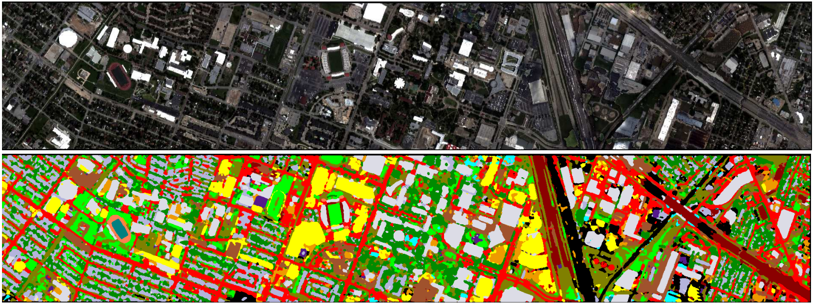
- Post processing: majority vote on
  - 5 independent runs
  - $5 \times 5$ moving window

## Results

| Exp. # | Spectral bands | Morphology on spectral | LiDAR | Morphology on LiDAR | NDVI | **Kappa statistic** |
|---|---|---|---|---|---|---|
| 1 | | | ✓ | | | 0.319 |
| 2 | | | ✓ | ✓ | | 0.702 |
| 3 | ✓ | | | | | 0.832 |
| 4 | ✓ | | ✓ | | | 0.868 |
| 5 | ✓ | | ✓ | | ✓ | 0.869 |
| 6 | ✓ | ✓ | | | | 0.899 |
| 7 | ✓ | ✓ | ✓ | ✓ | | 0.942 |
| 8 | ✓ | ✓ | ✓ | ✓ | ✓ | **0.946** |

[Matasci et al., Hyperspectral and LiDAR data fusion for high resolution urban land cover/land use classification, Swiss Geoscience Meeting, 2013]

## Results (exp #8)

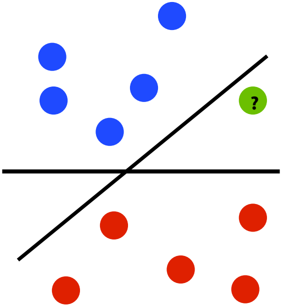


- grass_healthy
- grass_stressed
- grass_synthetic
- tree
- soil
- water
- residential
- commercial
- road
- highway
- railway
- parking_lot_empty
- parking_lot_vehicles
- tennis_court
- running_track

[Matasci et al., Hyperspectral and LiDAR data fusion for high resolution urban land cover/land use classification, Swiss Geoscience Meeting, 2013]

## But we are still tributary of training specificities

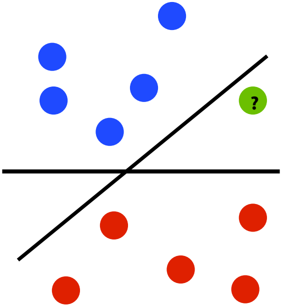
- There are physical facts that we know and we want to be invariant to!
  *e.g. rotation, shadowing, scaling of objects*




- The example assumes invariance to horizontal transformations
- Given the training data, the point ? is hard to classify
- Modify the SVM to incorporate prior knowledge

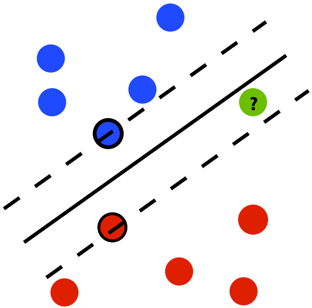
## But we are still tributary of training specificities

- There are physical facts that we know and we want to be invariant to!
  *e.g. rotation, shadowing, scaling of objects*

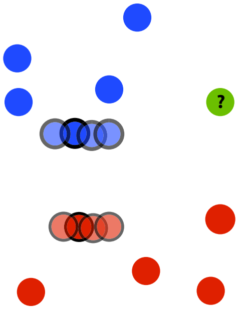



- The example assumes invariance to horizontal transformations
- Given the training data, the point ? is hard to classify
- Modify the SVM to incorporate prior knowledge


University of Zurich[UZH]

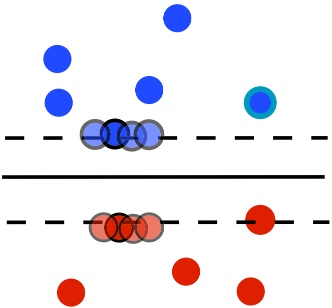



Step 1 Train a SVM and find the SVs




Step 1 Train a SVM and find the SVs

Step 2 VSVs: perturbate SVs to which the solution should be invariant



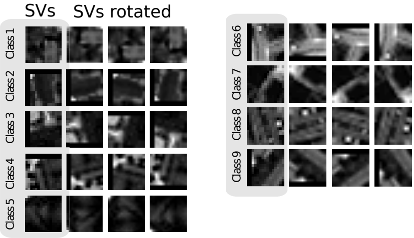
Step 1 Train a SVM and find the SVs

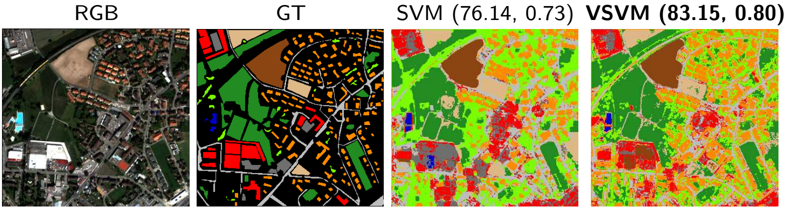
Step 2 VSVs: perturbate SVs to which the solution should be invariant

Step 3 Train a SVM with both SVs and VSVs

## Example: encoding invariance to rotations:

- Quickbird image + 18 spatial features
- Size: 329 × 347 pixels
- 9 classes
- VSVM encodes invariance to rotation!




- Both classifiers show high classification scores
- VSVM improves classification score over +7%
- VSVM is however more computationally demanding

[Izquierdo-Verediguer et al., Encoding Invariances in Remote Sensing Image Classification With SVM, GRSL 2013]

## Summary for classification

- Multi- and Hyperspectral image classification are challenging problems
- High dimensional feature spaces scarcely populated!
- Statistical approaches:
    - Supervised algorithms
    - Semisupervised algorithms
- Kernel methods are the current state-of-the-art classifiers
- More info in the classifiers implies improved signal model
    - More samples (by sampling or synthesizing)
    - More meaningful features
    - Multitemporal information
    - More concurrent sensors

# Part 4: Conclusions, source code and resources

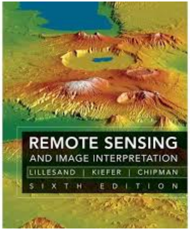
# Summary

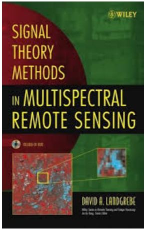
- Today, we introduced machine learning for remote sensing image processing
- We focused on two major tasks: **image classification** and **feature extraction**

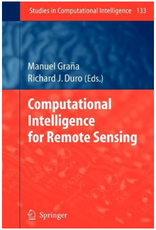
- We reviewed the basis and provided some MATLAB scripts to try them out.

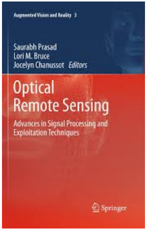
- Need more resources?
  - **Camps-Valls et al.** 'Advances in Hyperspectral Image Classification', IEEE Signal Processing Magazine, 31: 45-54, 2014.
  - **Gomez-Chova et al.**, 'Multimodal Classification of Remote Sensing Images: A Review and Future Directions', Proceedings of the IEEE, 103, 1560-1584, 2015.

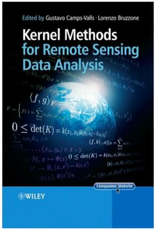
## Some relevant books:

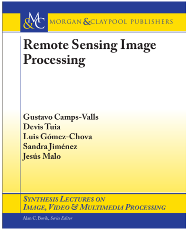


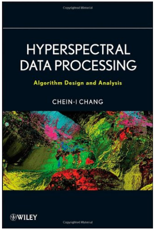



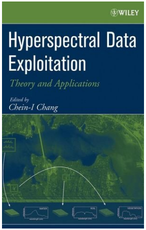



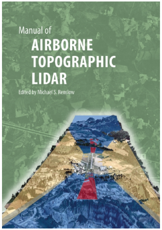



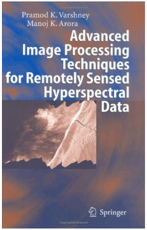



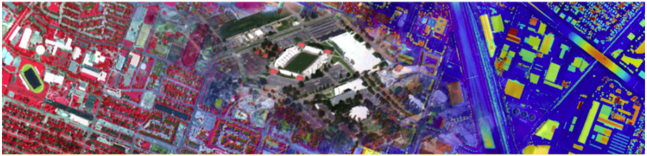



University of Zurich[UZH]

## Need data?

The Image Analysis and Data Fusion Technical Committee of the IEEE hosts a yearly Data Fusion Contest. Data and info can be found on the **IADF website**.

http://www.grss-ieee.org/community/technical-committees/data-fusion/data-fusion-contest/



>> 2013: LiDAR point clouds + hyperspectral aerial data


>> 2014: thermal hyperspectral + RGB VHR data

# Thank you for listening!

devis.tuia@geo.uzh.ch

Greatest thanks to Gustau Camps-Valls, with whom we prepared the slides 

Supported by:

FNSNF
SWISS NATIONAL SCIENCE FOUNDATION



University of Zurich[UZH]