S4HDD

Silviu Filote

July 2023

Contents

1 Lesson 1

1.1 Spatial model

$$y(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s})' \cdot \boldsymbol{\beta} + \varepsilon(\boldsymbol{s})$$

Osservazioni:

- $y(\boldsymbol{s})$ is the observation at generic spatial location $\boldsymbol{s} \in \mathbb{R}^2 = (latitude, longitude)$ or $\boldsymbol{s} \in \mathbb{S}^2$
- $\boldsymbol{x}(\boldsymbol{s})$ are generic spatial covariates, $\boldsymbol{x}(\boldsymbol{s_i})$ is the covariates in a specific location $\boldsymbol{s}_i$
- $\boldsymbol{\beta}$ is a vector of parameters, it has to be estimated
- $\varepsilon(\boldsymbol{s})$ is the random error at spatial location $\boldsymbol{s}$ (e.g. $\varepsilon(\boldsymbol{s}) \sim NID(0, \sigma^2_{\boldsymbol{\varepsilon}})$)
- $\mathbb{R}^2$ is the 2D space (plane) while $\mathbb{S}^2$ is the sphere embedded in $\mathbb{R}^3$
- n locations N data $\Rightarrow n = N$ (no multiple observations in the location, cause temporal)
- The data set has this form:

$$\big\{\big(\boldsymbol{x_1}(\boldsymbol{s_1}), y_1(\boldsymbol{s_1})\big), \ldots, \big(\boldsymbol{x_n}(\boldsymbol{s_n}), y_n(\boldsymbol{s_n})\big)\big\}$$

$$\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})', \quad i = 1, \ldots, n \quad p = covariates$$

1.2 Models residuals

$$\begin{aligned} e(t) = \varepsilon(\boldsymbol{s}) &= y(\boldsymbol{s}) - \boldsymbol{x}(\boldsymbol{s})' \cdot \hat{\boldsymbol{\beta}} \\ &= y(\boldsymbol{s}) - \hat{y}(\boldsymbol{s}) \end{aligned}$$

We check if residuals $\varepsilon(\boldsymbol{s})$ are IID which means:

- ***Are residuls spatially uncorrelated?*** Most of the time the residuals are correlated due to the fact that we don't have all the information in the dataset, so we are not observing all the covariates. The model presented before is a **suboptimal model**.
- ***Is the residul variance constant in space? (Homoscedasticity)*** Heteroscedasticity means that we have information left inside the residuals, thats because our model didnt capture all the variability.

Observations:

- In general, residuals $e(\boldsymbol{s})$ are ***spatially correlated***, because $\boldsymbol{x}(\boldsymbol{s})' \cdot \boldsymbol{\beta}$ cannot entirely capture the data variability. Probably this happens because measure all of the covariates can be expensive.
- When this happens we might (increase the complexity of the model):
  - Add covariates
  - Add transformations of covariates (polynomials)
  - Add interactions between covariates

$$\Rightarrow \textit{This may not be enough to have IID residuals}$$

What happens if we ignore the spatial correlation of residuals?

- The model fitting capability is lower than it should be
- Variance of estimators is wrong $\Rightarrow$ model assumptions is that $\varepsilon(\boldsymbol{s})$ is IID and normally distributed
- Confidence intervals might have significance levels lower than their nominal levels
- Spatial predictions are poor (especially in model validation)
- We can't trust our model and our predictions

1.3 Spatial model with latent variable

$$y(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta} + \alpha w(\boldsymbol{s}) + \varepsilon(\boldsymbol{s})$$

Observations:

- $w(\boldsymbol{s})$ is a ***latent (non observed) random variable*** spatially correlated with unitary variance
- $w(\boldsymbol{s})$ is included as a component in the model to capture the spatial variation that is not explained by the observed covariates $\boldsymbol{x}(\boldsymbol{s})$. Including $w(\boldsymbol{s})$ in the model, you acknowledge the presence of spatial correlation that cannot be explained by the covariates alone.
- $w(\boldsymbol{s})$ models the missing information and/or what the model doesnt capture
- $\varepsilon(\boldsymbol{s})$ models the measurement error + model error
- $w(\boldsymbol{s}) \perp \varepsilon(\boldsymbol{s})$, or $cov\big(w(\boldsymbol{s}), \varepsilon(\boldsymbol{s})\big) = 0$
- α is a scale parameter to be estimated $\Rightarrow$ it is like a weight
- $corr\big(w(\boldsymbol{s}), w(\boldsymbol{s'})\big) = \rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$
- $\rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$ is a bivariate function (on $\boldsymbol{s}$ and $\boldsymbol{s}'$) with unknown parameter vector $\boldsymbol{\theta}$
- Conditionally on $w(\boldsymbol{s})$, observed data $y_i(\boldsymbol{s}_i)$ are realizations of mutually independent random variables Y_i with conditional mean $E(Y_i|w) = \boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta} + \alpha w(\boldsymbol{s})$ and conditional variance σ^2_ε

Insights

- The latent variable $w(\boldsymbol{s})$ is a random variable that **is spatially correlated**. In other words, the value of $w(\boldsymbol{s})$ at one location $\boldsymbol{s}$ is related to the values of w at nearby locations. This spatial correlation is quantified by a spatial correlation function, typically denoted as $\rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ represents the parameters of this function.

$$corr\big(w(\boldsymbol{s}), w(\boldsymbol{s'})\big) = \rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$$

- The spatial correlation function, $\rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$, describes how the correlation between $w(\boldsymbol{s})$ and $w(\boldsymbol{s}')$ changes as a function of the spatial separation between locations $\boldsymbol{s}$ and $\boldsymbol{s}'$. It depends on the parameter vector $\boldsymbol{\theta}$, which needs to be estimated.
- The spatial variation in $w(\boldsymbol{s})$ is determined by the spatial correlation function $\rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$. Locations that are closer to each other will tend to have more similar values of $w(\boldsymbol{s})$ due to the positive correlation described by the spatial correlation function.
- It's important to note that, conditionally on $w(\boldsymbol{s})$, the observed data $y_i(\boldsymbol{s}_i)$ are assumed to be mutually independent random variables. This means that the correlation in $w(\boldsymbol{s})$ does not directly affect the correlation in the residuals $\varepsilon(\boldsymbol{s})$.
- $w(\boldsymbol{s})$ is spatially varying (in general each location $\boldsymbol{s}$ has a differen $w(\boldsymbol{s})$ value)
- ***How overfitting is avoided?*** we can't use $w(\boldsymbol{s})$ in order to estimate the exact value of $y(\boldsymbol{s})$

$$\underset{10=}{y(\boldsymbol{s})} = \underset{7}{\boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta}} + \underset{-2}{\alpha w(\boldsymbol{s})} + \underset{5}{\varepsilon(\boldsymbol{s})} \qquad \Rightarrow \qquad overfitting$$

- The correlation function $\rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta})$ can be any positive-definite function. This condition imposes that the linear combination:

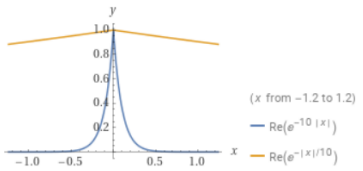
$$\sum_{i=1}^{m} \alpha_i \cdot w(\boldsymbol{s}_i) \qquad has\ positive\ variance$$

- In most cases $\rho(\boldsymbol{s}, \boldsymbol{s}', \boldsymbol{\theta}) = \rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta}) = \rho(u; \boldsymbol{\theta})$, where $||\cdot||$ is the distance (Euclidean or geodetic) between $\boldsymbol{s}$ and $\boldsymbol{s}'$
- In this case the spatial correlation only **depends on the distance** u and not on the coordinates.
- Instead if the correlation **depends on the coordinates**: $w(0)$ and $w(1)$ would be different from the correlation between $w(1)$ and $w(2)$

1.4 Exponential spatial correlation function

$$\rho(u;\theta) = exp\left(-\frac{|\boldsymbol{s}-\boldsymbol{s}'|}{\theta}\right) = exp\left(-\frac{u}{\theta}\right)$$

θ fixed, depends on distance u



Observations:

- This function describes how the spatial correlation between two locations decreases exponentially as the distance between them, denoted as u increases.
- $\theta > 0$ is a scalar value. The larger the value of θ, the slower the correlation decreases with distance, and the larger the effective range of spatial dependence. Conversely, a smaller value of θ results in a faster decay of correlation with distance and a shorter effective range.
- The exponential correlation function is **simple** and computationally tractable, making it a common choice in spatial modeling. However, its simplicity can also be a limitation because it may **not be flexible enough** to capture complex spatial correlation patterns that are observed in some datasets.
- The suitability of the exponential correlation function depends on the specific characteristics of the spatial data under consideration. It is particularly appropriate when there is a clear spatial decay in correlation with distance, and the range of spatial dependence is relatively short. In cases where the spatial correlation pattern is not well described by exponential decay, alternative correlation functions (e.g., Gaussian, Matérn, or spherical) may be more appropriate.

1.5 Matérn spatial correlation function

$$\rho(u;\theta) = \{2^{K-1}\Gamma(K)\}^{-1}\left(\frac{u}{\phi}\right)K_K\left(\frac{u}{\phi}\right)$$

Observations:

- K_K: is the modified Bessel function of the second kind of order K
- K: The parameter K is known as the smoothness or differentiability parameter. It determines how smooth or rough the spatial correlation function is. It is a positive real number $K > 0$.
- u: This is the separation distance between two spatial points for which you want to compute the correlation. It is a non-negative real number.
- ϕ: with $\phi > 0$ is the range parameter. It controls how quickly the spatial correlation decreases with increasing separation distance. A smaller ϕ leads to a more rapid decrease in correlation with distance.

1.6 Gaussian process (GP)

- $w(\boldsymbol{s})$ not directly observed but can be inferred from the observed data set $\{(\boldsymbol{x_1}(\boldsymbol{s_1}), y_1(\boldsymbol{s_1})), \ldots, \}$
- First, the distribution of $w(\boldsymbol{s})$ must be specified in order to estimate it from the dataset
- Formally we want that $w(\boldsymbol{s})$ is a zero mean Gaussian process (GP) with unitary variance and spatial correlation function $\rho(u;\boldsymbol{\theta})$
- $w(\boldsymbol{s})$ is a gaussian process if the joint distribution of $w(\boldsymbol{s}_1), \ldots, w(\boldsymbol{s}_n)$ is multivariate normal (see SMS2) for any integer n and any set of locations $(\boldsymbol{s_1}, \ldots, \boldsymbol{s_n})$
- Simulations are useful to understand if a given correlation function is suitable to model the observed data set
- A GP is continuous in space. This means that:
  - It can be simulated on a regular grid or
  - On a irregular grid
  - For any given number of spatial locations
- When the GP is simulated on a regular grid, the simulated values referes to the centres of the pixels

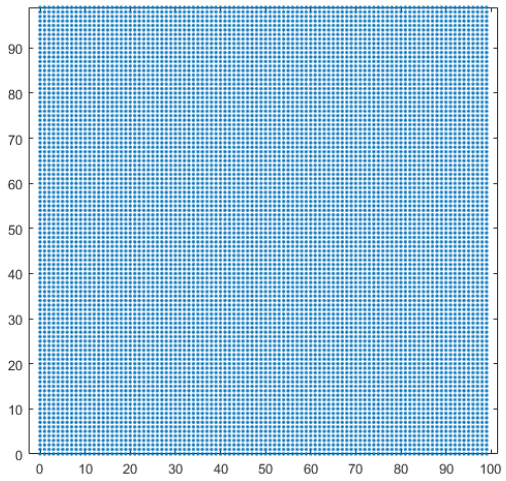
1.7 Matlab exercises 2-3

1.8 Spatial correlation - Regular grid

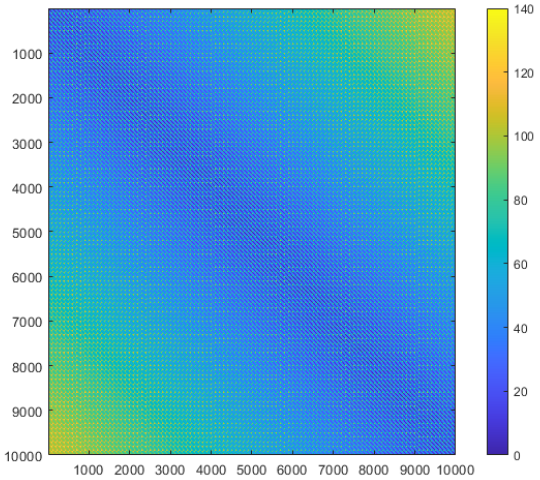
```matlab
% Vettore coordinate
x = 0:1:99;
y = 0:1:99;

% Create a grid of coordinates (X, Y) using 'x' and 'y'vectors.
[X,Y] = meshgrid(x,y);

% Convert the grid coordinates (X, Y) into a 2D array 'coord'.
coord = [X(:) Y(:)];

% Plot the coordinates as points.
% Set the aspect ratio of the plot to be equal.
plot(coord(:,1),coord(:,2),'.');
axis equal

% Set a correlation length parameter.
theta = 0.1;

% Calculate pairwise Euclidean distances between points.
% distance matrix
dist = pdist2(coord,coord,"euclidean");
```

```matlab
% Display the distance matrix as an image.
imagesc(dist);

% Calculate the spatial correlation using the exponential
% correlation function.
sp_corr = exp(-dist/theta);

% Display the spatial correlation matrix as an image.
imagesc(sp_corr);

% GP simulation:
% Generate a sample from a multivariate Gaussian distribution.
v = mvnrnd(zeros(1,size(dist,1)),sp_corr,1);

% Reshape the sample to match the grid shape.
v = reshape(v,size(X));

imagesc(v)            % Display the generated GP sample as an image.
clim([-3,3])          % Set the color limit for the plot.
colorbar              % Add a color bar to the plot.
```

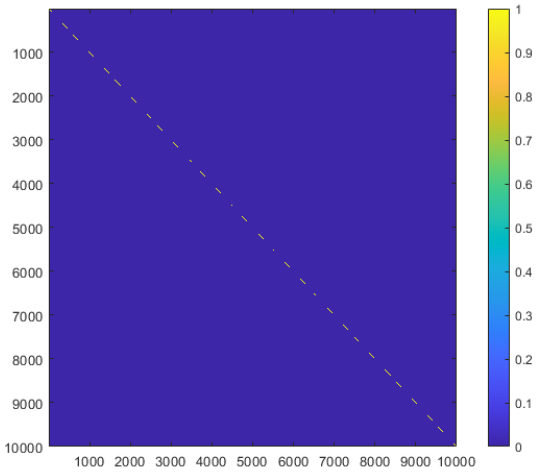
Uniform grid of the vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$



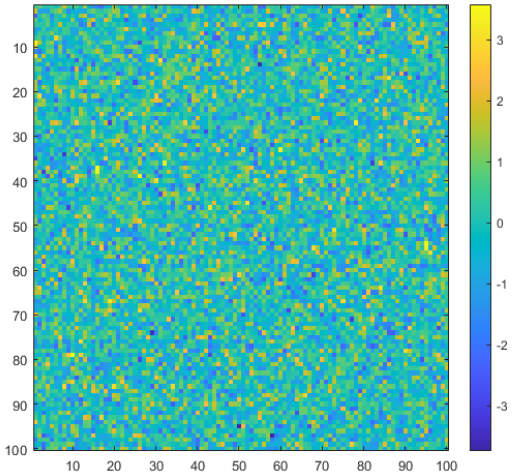
Distances matrix



Spatial correlation with $\theta = 0.1$



GP sample as an image

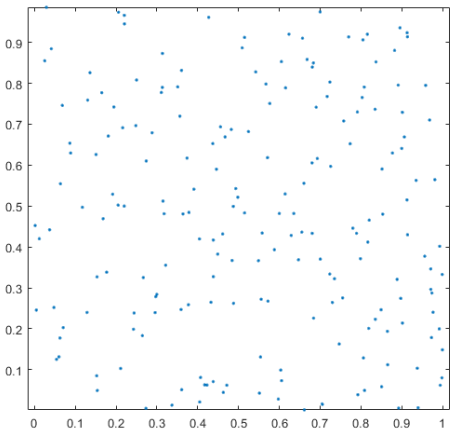


1.9 Spatial correlation - Irregular grid

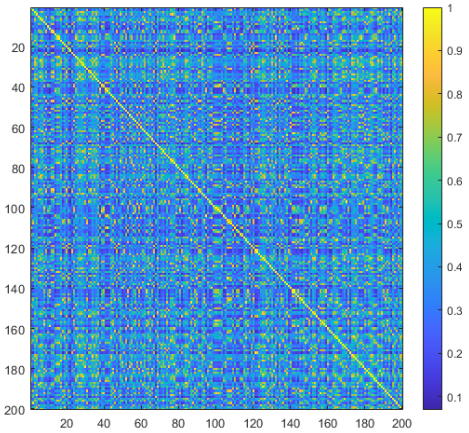
```
% Generate 200 random points in a 2D space with uniform coordinates
% between 0 and 1.
coord = unifrnd(0,1,200,2);

% Plot the irregularly spaced points.
plot(coord(:,1),coord(:,2),'.');

% Set the aspect ratio of the plot to be equal.
axis equal

% Set a correlation length parameter.
theta = 0.5;
```

```
% Calculate pairwise Euclidean distances between points.
dist = pdist2(coord,coord,"euclidean");

% Display the distance matrix as an image.
imagesc(dist);

% Calculate the spatial correlation using the exponential
% correlation function
sp_corr = exp(-dist/theta);

% GP simulation:
% Generate a sample from a multivariate Gaussian distribution.
v = mvnrnd(zeros(1,size(dist,1)),sp_corr,1);

% Scatter plot with color-coded values based on the GP sample.
scatter(coord(:,1),coord(:,2),40,v,"filled")
clim([-3,3])
colorbar
```

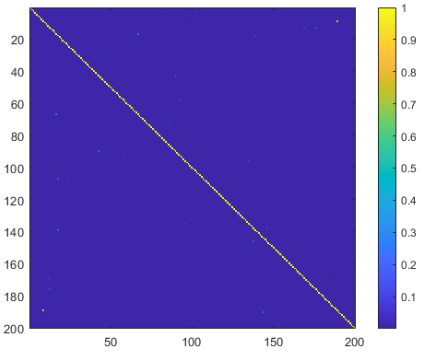
Irregular grid of the vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$



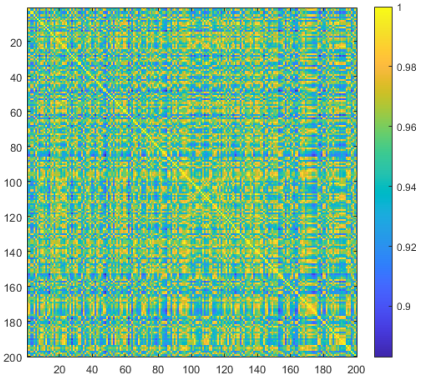
Distances matrix



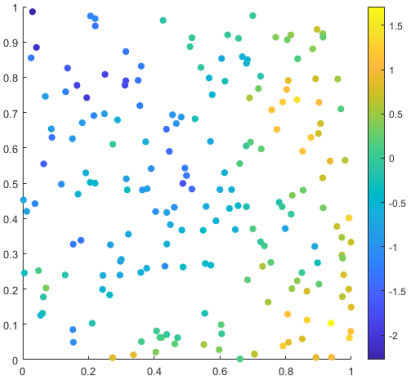
Spatial correlation with $\theta = 0.006$



Spatial correlation with $\theta = 10.5$



GP sample as an image



2 Lesson 2

2.1 Spatial model with latent variable

$$y(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta} + \alpha w(\boldsymbol{s}) + \varepsilon(\boldsymbol{s}) \tag{1}$$

Remarks:

- $w(\boldsymbol{s}) \sim GP\big(0, \rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta})\big)$, is an approximation, in IRL we don't really know its distribution. Assuming that means if there is etheroschedasticity the variance not captured is inside the ε
- $w(s)$ is a gaussian process and its not a normal multivariate in fact the mean is a scalar and the **variance** is not a matrix but is a **function** related to the correlation function and parameterized by $\boldsymbol{\theta}$
- where $||\boldsymbol{s} - \boldsymbol{s}'||$ is the euclidean distance between 2 generics points
- $\rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta}) = corr\big(w(\boldsymbol{s}), w(\boldsymbol{s'})\big)$
- $\varepsilon(\boldsymbol{s}) \sim N(0, \sigma^2_\varepsilon)$, we can use the normal because there is no correlation in space and the variables are indipendent
- The unknown parameter set is $\Psi = \{\boldsymbol{\beta}, \alpha, \sigma^2_\varepsilon, \boldsymbol{\theta}\}$
- How to estimate Ψ from data? **MLE → EM**

Remarks: $w(\boldsymbol{s}) \sim GP$ is an approximation, we dont really know what is its distribution. In fact if we impose that $w(s) \sim GP$ and $w(s)$ is not a gaussin process, $\boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta}$ are not capable to explain well the data observed and the **error** is in $\varepsilon(\boldsymbol{s})$. We can confirm this if:

- there si spatial correlation left inside $\varepsilon(\boldsymbol{s})$.
- or $\varepsilon(\boldsymbol{s})$ is etheroschedastic in space

2.2 Maximum likelihook estimate

- We rely on MLE to estimate the model parameter vector Ψ, because MLE has good properties and in case we can use the EM algorithm (if needed).
  - **Correct Model Specification:** the model you choose should be a valid representation of the data-generating process. If the model is misspecified, MLE estimates may not be accurate.
  - **the observations must be IID:** assumes that the data points are drawn from the same probability distribution with the same parameter values and the observations are independent of each other. This means that the probability of one observation does not depend on the values of other observations.
  - **Finite Parameter Space:** MLE assumes that the parameter space is finite. This means that the parameter values being estimated exist within a specific, well-defined range.
  - We use **the Expectation-Maximization (EM)** algorithm when the data has missing or latent variables.
- with EM the missing values can be:
  - $y(\boldsymbol{s})$ but if we have a spatial dataset we just delete that information, instead in spatial temporal dataset we dont delete the y(t) but assign it a NA value
  - or $w(s)$
- Likelihood function of (3.2) is

$$L(\Psi; \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{X}) = L(\Psi; \boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X}) L(\Psi; \boldsymbol{w})$$

  - $\boldsymbol{y} = (y_1, \ldots, y_n)'$ is the vector of observations at n spatial locations
  - $\boldsymbol{w} = (w_1, \ldots, w_n)'$ is the vector of latent variables at n spatial locations ⇒ **we pretend for the moment to observe it**
  - $\boldsymbol{X}$ is the $n \times p$ design matrix, with p covariates
- we **assume** that the covariates does not influence the $w(s)$, the covariates are numbers without uncertainty

2.3 Likelihood decomposition

$$\begin{aligned} L(\Psi; \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{X}) &= L(\Psi; \boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X}) L(\Psi; \boldsymbol{w}) \\ &= L(\boldsymbol{\beta}, \alpha, \sigma_\varepsilon^2; \boldsymbol{y}|\boldsymbol{x}, \boldsymbol{X}) L(\boldsymbol{\theta}; \boldsymbol{w}) \end{aligned}$$

Remarks:

- $L(\Psi; \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{X})$ is the complete-data likelihood (which assumes $\boldsymbol{w}$ to be known)
- In reality I dont observed $\boldsymbol{w}$ so Im forced to do this decomposition
- $L(\boldsymbol{\beta}, \alpha, \sigma_\varepsilon^2; \boldsymbol{y}|\boldsymbol{x}, \boldsymbol{X})$ and $L(\boldsymbol{\theta}; \boldsymbol{w})$ are described by a normal n-variate distributions. This derive from the fact that:

$$w(s) \sim GP \ \ and \ \varepsilon(s) \sim N$$

2.4 Log-likelihood function

$$-2log(L_\Psi) = log|\Sigma_\varepsilon| + \boldsymbol{e}'\Sigma_\varepsilon \boldsymbol{e} + log|\Sigma_w| + \boldsymbol{w}'\Sigma_w^{-1}\boldsymbol{w}$$

Remarks:

- As usual we prefer to work with $log(L_\Psi) \rightarrow$ maximize
- $-2log(L_\Psi) \rightarrow$ minimize
- $\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \alpha\boldsymbol{w}$, error vector (we still pretend to know $\boldsymbol{w}$)
- $\Sigma_\varepsilon = \sigma_\varepsilon^2 \boldsymbol{I}_n$, with $\boldsymbol{I}_n$ the identity matrix of dimension $n \rightarrow$ diagonal matrix
- Σ_ε we don't have spatial correlation that means ε is indipendent so we only have a diagonal matrix with the same value on the diagonal and all the coviances are 0 (omoschedasticity, same variance between each point in space $\rightarrow \sigma_\varepsilon^2$)
- Σ_w is the $n \times n$ correlation matrix (e.g. $exp(-D/\theta)$, with D the distance matrix)
- If we know $\boldsymbol{w}$ we can minimize direclty Log-likelihood (only one unknown)

2.5 ML estimate

$$\hat{\Psi} = argmin_{\boldsymbol{\beta}, \alpha, \sigma_\varepsilon^2, \boldsymbol{\theta}} = log|\Sigma_\varepsilon| + \boldsymbol{e}'\Sigma_\varepsilon \boldsymbol{e} + log|\Sigma_w| + \boldsymbol{w}'\Sigma_w^{-1}\boldsymbol{w}$$

Remarks:

- Argmin because we are considering $-2log(L_\Psi)$
- Unfortunately minimizing $-2log(L_\Psi)$ is not feasible (not computationally applicable), plus ***w is latent and not observed (we cant calcolate this parts)*** $\Rightarrow$ we must rely on the EM algorithm

2.6 EM algorithm

- The EM is an iterative algorithm for MLE and under some assumptions the estimate of EM is the same as the ML estimate $\Rightarrow$ this is the reason why we use it
- First iteration starts with fixed initial values choosed by us $\hat{\Psi}^0$ (usually given by OLS and method of moments). If we inizialize $\hat{\Psi}^0$ with parameters that are really far from the optimal there is the risk to converge to **local minimum**
- EM algorithm has two steps: Expectation and Maximization

E-step:

$$Q(\Psi, \hat{\Psi}^m) = \mathbb{E}_{\hat{\Psi}^m}\big[-2logL(\Psi; \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{X}|\boldsymbol{y})\big]$$

Closed form:

$$\begin{aligned} Q(\Psi, \hat{\Psi}^m) &= \mathbb{E}_{\hat{\Psi}^m}\big[-2logL(\Psi; \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{X}|\boldsymbol{y})\big] \\ &= tr\big[\Sigma_\varepsilon^{-1}\big(\mathbb{E}(\boldsymbol{e}|\boldsymbol{y})\mathbb{E}(\boldsymbol{e}|\boldsymbol{y})' + Var(\boldsymbol{e}|\boldsymbol{y})\big)\big] \\ &+ tr\big[\Sigma_\varepsilon^{-1}\big(\mathbb{E}(\boldsymbol{w}|\boldsymbol{y})\mathbb{E}(\boldsymbol{w}|\boldsymbol{y})' + Var(\boldsymbol{w}|\boldsymbol{y})\big)\big] \end{aligned}$$

Remarks:

1. $\mathbb{E}(\boldsymbol{e}|\boldsymbol{y}) = \boldsymbol{y} - \boldsymbol{X\beta} - \alpha\mathbb{E}(\boldsymbol{w}|\boldsymbol{y})$
2. $Var(\boldsymbol{e}|\boldsymbol{y}) = Var(\boldsymbol{y} - \boldsymbol{X\beta} - \alpha\boldsymbol{w}|\boldsymbol{y}) = \alpha^2 Var(\boldsymbol{w}|\boldsymbol{y})$
3. $\mathbb{E}(\boldsymbol{w}|\boldsymbol{y}) = Cov(\boldsymbol{w}, \boldsymbol{y})Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X\beta}]$
4. $Var(\boldsymbol{w}|\boldsymbol{y}) = \Sigma_w - Cov(\boldsymbol{w}, \boldsymbol{y})Var(\boldsymbol{y})^{-1}Cov(\boldsymbol{w}, \boldsymbol{y})'$
5. $Var(\boldsymbol{y}) = Var(\boldsymbol{X\beta} + \alpha\boldsymbol{w} + \boldsymbol{\varepsilon}) = Var(\alpha\boldsymbol{w} + \boldsymbol{\varepsilon})$
   $= \alpha^2 Var(\boldsymbol{w}) + Var(\boldsymbol{\varepsilon}) + 2Cov(\boldsymbol{w}, \boldsymbol{\varepsilon})$
6. $Var(\boldsymbol{w}) = \Sigma_w$
7. $Var(\boldsymbol{\varepsilon}) = \Sigma_\varepsilon = \sigma_\varepsilon^2 \boldsymbol{I}_n$
8. $2Cov(\boldsymbol{w}, \boldsymbol{\varepsilon}) = \boldsymbol{0}$
9. $Cov(\boldsymbol{w}, \boldsymbol{y}) = Cov(\boldsymbol{w}, \boldsymbol{X\beta} + \alpha\boldsymbol{w} + \boldsymbol{\varepsilon}) = Cov(\boldsymbol{w}, \alpha\boldsymbol{w})$
   $= \alpha Cov(\boldsymbol{w}, \boldsymbol{w}) = \alpha Var(\boldsymbol{w}) = \alpha\Sigma_w$

M-step (Maximization):

$$\Psi^{\langle m+1\rangle} = argmax_\Psi\, Q\big(\Psi, \Psi^{\langle m\rangle}\big)$$

$$\frac{dQ\big(\Psi, \Psi^{\langle m\rangle}\big)}{d\Psi} = 0$$

$$\alpha^{\langle m+1\rangle} = \frac{tr\big[(\boldsymbol{y} - \boldsymbol{X\beta}^{\langle m\rangle})E(\boldsymbol{w}|\boldsymbol{y})'\big]}{tr[E(\boldsymbol{w}|\boldsymbol{y})E(\boldsymbol{w}|\boldsymbol{y})' + Var(\boldsymbol{w}|\boldsymbol{y})]}$$

$$\boldsymbol{\beta}^{\langle m+1\rangle} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\Big[\boldsymbol{X}'\Big(\boldsymbol{y} - \alpha^{\langle m+1\rangle}E(\boldsymbol{w}|\boldsymbol{y})\Big)\Big]$$

$$\sigma_\varepsilon^{2\langle m+1\rangle} = \frac{1}{n}tr[E(\boldsymbol{e}|\boldsymbol{y})E(\boldsymbol{e}|\boldsymbol{y})' + Var(\boldsymbol{e}|\boldsymbol{y})]$$

$$\boldsymbol{\theta}^{\langle m+1\rangle} = argmin_{\boldsymbol{\theta}}\ \ log\,|\Sigma_w^{-1}(\boldsymbol{\theta})| + tr[\Sigma_w^{-1}(\boldsymbol{\theta})(\widehat{\boldsymbol{w}}\widehat{\boldsymbol{w}}')]$$

$$\text{Where } \widehat{\boldsymbol{w}} = E_{\boldsymbol{\theta}^{\langle m\rangle}}(\boldsymbol{w}|\boldsymbol{y}) = E(\boldsymbol{w}|\boldsymbol{y})$$

Importante:

- (1) Il valore atteso di un vettore di numeri $\boldsymbol{y}$ sono i numeri stessi e il valore atteso si riduce su $\boldsymbol{w}$. Sto cercando di ottenere qual'é il valore atteso della variabile latente $\boldsymbol{w}$ condizionatamente ai dati che ho osservato $\boldsymbol{y}$.
- (2) La varianza condizionata di un scalare é 0 e sopravvive la var su $\boldsymbol{w}$. Fare l'inversa di una matrice é computazionalemnte oneroso $Var(\boldsymbol{y})^{-1}$
- (3) rappresenta la stima della variabile latente. Se la $Var(\boldsymbol{y}) \to \infty$, ossia l'errore di misura é molto alto $Var(\varepsilon)$ non posso fidarmi di $[\boldsymbol{y} - \boldsymbol{X\beta}]$ (errore di modello) perché é tutto errore e dunque la stima di $\boldsymbol{w}$ é 0, al contrario se $Var(\boldsymbol{y}) \to Var(\boldsymbol{w})$, dunque non abbiamo errore di misura e quindi nemmeno errore di modello e la stima di $\mathbb{E}(\boldsymbol{w}|\boldsymbol{y}) = \boldsymbol{y} - \boldsymbol{X\beta}$

$Var(\varepsilon) = \infty$	$Var(y) \to \infty$	$\mathbb{E}(\boldsymbol{w}\|\boldsymbol{y}) = 0 = mean\ GP$
$Var(\varepsilon) = 0$	$Var(y) \to Var(\boldsymbol{w})$	$\mathbb{E}(\boldsymbol{w}\|\boldsymbol{y}) = \boldsymbol{y} - \boldsymbol{X\beta}$

- (4) viene interpretata come incertezza

$Var(\varepsilon) = \infty$	$Var(y) \to \infty$	$Var(\boldsymbol{w}\|\boldsymbol{y}) = \Sigma_w$
$Var(\varepsilon) = 0$	$Var(y) \to Var(\boldsymbol{w})$	$Var(\boldsymbol{w}\|\boldsymbol{y}) = \Sigma_w - \Sigma_w = 0$

- (7) Matrice delle distanze é deterministica, $\boldsymbol{\theta}$ é un parametro
- (8) Assunzione del modello, le variabili tra di loro sono indipendenti
- (9) $Cov(\boldsymbol{w}, X\beta) = 0$ perché $\boldsymbol{X}$ costante, $Cov(\boldsymbol{w}, \varepsilon) = 0$, rimane $Cov(\boldsymbol{w}, \alpha\boldsymbol{w}) = 0$
- (α^{m+1}) dipende dalla (3) é possono esserci problemi di identificabilitá, var alta α non ha significato
- ($\boldsymbol{\beta}^{m+1}$) aggiornamento di beta, togliamo alpha appena stimato e il valore atteso di $\boldsymbol{w}$, perché non voglio avere l'effetto di $\boldsymbol{w}$ su $\boldsymbol{\beta}$

- $(\sigma_\varepsilon^{m+1})$ il residuo ora é $\boldsymbol{e|y} = \boldsymbol{y} - \boldsymbol{X\beta} - \alpha\mathbb{E}(\boldsymbol{w|y})$
- $(\boldsymbol{\theta}^{m+1})$ é l'unica quantitá che non riesco a stimare in forma chiusa, infatti questa stima é data da una minimizzazione della verosimiglianza che riguarda solo theta. Il minimizzatore cambia theta internamente e ogni volta che lo fa cambiano anche le matrici coinvolte ⇒ **penalizzazione computazionale molto elevata**
- Se ho dei dubbi che la mia stima del EM non sia quella della ML, posso randomizzare i valori inziali del mio $\hat{\Psi}^0$ e vedo se l'algoritmo converge alle stesse soluzioni
- Algoritmo EM non fornisce direttamente alcuna incertezza sui parametri stimati: $\alpha, \boldsymbol{\beta}, ..$ Mi da solo l'incertezza della variabile lantente $Var(\boldsymbol{w|y})$
- Solitamente l'incertezza sui parametri tende ad essere molto bassa, tranne sul $(\boldsymbol{\theta}^{m+1})$, questo perché risulta essere poco identificabile, anche raddoppiandolo la verosimiglianza cambia poco
- $\boldsymbol{w}$ viene compensato da $\boldsymbol{X\beta}$, quindi anche cambiandolo é il modello che mi da una mano
- Come da variogramma: θ piú é grande piú c'é correlazione spaziale, se dunque mando $\theta \to \infty$ ho $\boldsymbol{w}$ costante, ossia la correlazione spaziale é cosí forte che indipendentemente dalla distanza $\boldsymbol{w}$ rimane costante

2.7 Variogram

- In spatial statistics, a variogram (also known as a semivariogram) is a graphical and mathematical tool used to quantify spatial dependence or spatial autocorrelation in a set of spatial data
- Spatial dependence refers to the idea that the values of a variable at one location are related to the values of the same variable at nearby locations. The variogram helps to characterize the spatial structure and variability of a spatial process.
- The variogram is typically calculated using pairs of observations at different locations within a study area. **The variogram measures the variability between pairs of points as a function of the distance or lag separating them.**
- The variogram is denoted as $2\gamma(\boldsymbol{s}, \boldsymbol{s'})$, instead the semivariogram is half the variogram $\gamma(\boldsymbol{s}, \boldsymbol{s'})$.

  The semivariogram was defined as half the average squared difference between the values at points ($\boldsymbol{s_1}$ and $\boldsymbol{s_2}$) separated at distance $\boldsymbol{h}$, where $\boldsymbol{M}$ is a point in the geometric field $\boldsymbol{V}$ and $\boldsymbol{f(M)}$ is the value at that point. The triple integral is over 3 dimensions, instead $\boldsymbol{h}$ is the separation distance (meters or km).

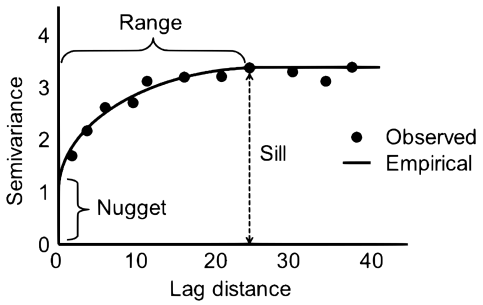
  Example: *the value $f(M)$ could rapresent the iron content in soil at some location M (with geographic coordinates of latitude, longitude, and elevation) over some region V with element of volume dV. To obtain the semivariogram for a given $\gamma(h)$, all pairs of points at that exact distance would be sampled. In practice it is impossible to sample everywhere, so the empirical variogram is used instead.*

  The variogram is twice the semivariogram and can be defined, equivalently, as the **variance** of the difference between field values at two locations ($\boldsymbol{s_1}$ and $\boldsymbol{s_2}$, note change of notation from M to $\boldsymbol{s}$ and f to Z) across realizations of the field

$$\textit{semivariogram:} \qquad \gamma(h) = \frac{1}{2V}\int\int\int_V \big[f(M+h) - f(M)\big]^2 dV$$

$$\textit{variogram:} \qquad 2\gamma(h) = var\bigg(\big(Z(\boldsymbol{s_1}) - \mu\big) - \big(Z(\boldsymbol{s_2}) - \mu\big)\bigg)^2$$

- The variogram is then plotted as a function of lag distance, showing how the semivariance changes with increasing separation between points. The resulting variogram plot can provide insights into the spatial structure of the variable under consideration, helping analysts to understand patterns such as clustering or spatial trends.



Parameters:

- *nugget n:* represents the discontinuity or the "jump" of the semivariogram at the origin (lag distance equals zero). It accounts for the variability at very short distances that might be due to measurement errors or other factors not captured by the spatial model. A higher nugget indicates a greater level of variability at very short distances.
- *sill s:* The sill is the limiting value of the variogram as the lag distance tends to infinity. It represents the total variability or the maximum semivariance in the spatial process. A higher sill indicates a larger overall variability in the spatial process.
- *range r:* The distance in which the difference of the variogram from the sill becomes negligible. It signifies the distance at which spatial autocorrelation or dependence between data points is no longer significant. A longer range indicates that spatial correlation or similarity between points persists over larger distances.

Nugget	*Represents short-range variability or variability at very small distances*
Sill	*Represents the total variability or the maximum semivariance in the spatial process*
Range	*Represents the distance at which spatial correlation becomes negligible*

Empirical variogram

- Variogram is made on countinous data which in IRL we don't have so the empirical variogram is base on the given data we have
- Calculate the differences between all pairs of data points at different lag distances (distances between points) and for each lag distance, calculate the semivariance, which indicates how much values differ at a given distance
- Group the semivariance values into bins based on the lag distance and calculate the average semivariance within each bin
- Plot the lag distance on the x-axis and the average semivariance on the y-axis. The resulting plot is the empirical variogram, which provides insights into the spatial structure of the data.

Applications:

- It a model of the temporal/spatial correlation of the observed phenomenon
- The **experimental variogram** that is a visualisation of a possible spatial/temporal correlation and the **variogram model** that is further used to define the weights of the kriging function
- The experimental variogram is an empirical estimate of the covariance of a Gaussian process. As such, it may not be positive definite and hence not directly usable in kriging, without constraints or further processing.

3 Lesson 3

3.1 Spatial model with latent variable

$$y(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta} + \alpha w(\boldsymbol{s}) + \varepsilon(\boldsymbol{s}) \tag{2}$$

Remarks:

- $w(\boldsymbol{s}) \sim GP\big(0, \rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta})\big)$
- $\rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta}) = corr\big(w(\boldsymbol{s}), w(\boldsymbol{s'})\big)$
- $\varepsilon(\boldsymbol{s}) \sim N(0, \sigma^2_\varepsilon)$ measurement/model error
- $\boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta}$ are the regressors associated with their coefficients. $\boldsymbol{x}(s)$ are numbers with no variance and no distribution, instead $\boldsymbol{\beta}$ is the related coefficients estimated with EM.
- The unknown parameters $\Psi = \{\boldsymbol{\beta}, \alpha, \sigma^2_\varepsilon, \boldsymbol{\theta}\}$ is estimated using the EM algorithm

3.2 Prediction for one site s_i

$$\hat{y}(\boldsymbol{s_i}) = \boldsymbol{x}(\boldsymbol{s_i})'\hat{\boldsymbol{\beta}} + \hat{\alpha}\hat{w}(\boldsymbol{s_i})$$

Remarks:

- The above formula gives the prediction at spatial locations s_i, with $i = 1, \dots, n$
- Seems to be more like a **filtering** than a prediction. Measurement error is cutted off
- We are making the prediction in a specific space location s_i
- Monitoring stations in space → we want to estimate the pollutants concentration → regular grid. s_i will be a pixel of the grid, the center of the pixel
- $\hat{w}(s_i) = \mathbb{E}[w(s_i)|Y]$, we estimate $\hat{w}$ using the observations in the dataset Y
- $\hat{y}(s_i)$ the prediction in a specific location requires the regressors in that location $\boldsymbol{x}(\boldsymbol{s_i})'\hat{\boldsymbol{\beta}}$ if not we can't do the prediction. A way to avoid this is to interpolate across space and use the interpolated values as $\boldsymbol{x}(\boldsymbol{s_i})$ in the model. This is not the best approach because $\boldsymbol{x}(\boldsymbol{s_i})$ should be a value with no variance, so deterministic, the iterpolation will add and error to the model.

Important:

- First of all we need $\boldsymbol{x}(\boldsymbol{s})$ (at spatial covariates $\boldsymbol{s}$)
- $\mathbb{E}(\boldsymbol{w}|\boldsymbol{y}) = Cov(\boldsymbol{w}, \boldsymbol{y})Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X\beta}]$
- Similarly $\hat{w}(\boldsymbol{s}) = \mathbb{E}(w(\boldsymbol{s})|\boldsymbol{y}) = Cov(w(\boldsymbol{s}), \boldsymbol{y})Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X\beta}] \in \mathbb{R}$
- In practice the prediction $\hat{w}(\boldsymbol{s})$ depends on the vector $\boldsymbol{y}$ of all observations
- More in details:

$$\begin{aligned}
\hat{w}(\boldsymbol{s}) = \mathbb{E}(w(\boldsymbol{s})|\boldsymbol{y}) &= Cov(w(\boldsymbol{s}), \boldsymbol{y})Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X\beta}] \\
Cov(w(\boldsymbol{s}), \boldsymbol{y}) &= Cov(w(\boldsymbol{s})|\boldsymbol{X\beta} + \alpha\boldsymbol{w} + \boldsymbol{\varepsilon}) \\
&= Cov\big(w(\boldsymbol{s}), \alpha\boldsymbol{w}\big) \\
&= \alpha Cov\big(w(\boldsymbol{s}), \boldsymbol{w}\big) \in \mathbb{R}^{1\times n}
\end{aligned}$$

$$Cov\big(w(\boldsymbol{s}), w(\boldsymbol{s_i})\big) = exp\bigg(-\frac{||\boldsymbol{s} - \boldsymbol{s_i}||}{\theta}\bigg)$$

- $Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X\beta}]$ seen in the EM algorithm and do not depend on $w(\boldsymbol{s})$
- $Cov(\boldsymbol{w(s)}, \boldsymbol{y})$ can be seen as a weigth vector, so very far from the network this covariance calcolated by the exponential will be close to 0 and the prediction we are making will be as well close to 0

3.3 Spatial prediction for multiple sites S

$$\hat{\boldsymbol{w}}(\boldsymbol{S}) = \mathbb{E}(\boldsymbol{w}(\boldsymbol{S})|\boldsymbol{y}) = Cov(\boldsymbol{w}(\boldsymbol{S}), \boldsymbol{y}) Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}]$$

- For one site $\boldsymbol{s_i}$, we try to cover the region with the grid and we predict the pollutant concentration for each pixel of the grid, instead the prediciton for multiple sites, we collect all the locations into $\boldsymbol{S}$ and the prediction will be a vector. This time we are doing a multiple prediction on multiple points on the grid
- Also if we want we can do predictions on blocks of the grid and not on all the grid
- Prediction is done for multiple spatial locations $\boldsymbol{S} = \{\boldsymbol{s}_1, \dots, \boldsymbol{s}_M\}$
- $Cov\big(\boldsymbol{w}(\boldsymbol{S}), \alpha\boldsymbol{w}\big) = \alpha Cov\big(\boldsymbol{w}(\boldsymbol{S}), \boldsymbol{w}\big)$
- $\alpha Cov\big(\boldsymbol{w}(\boldsymbol{S}), \boldsymbol{w}\big)$ is a $M \times n$ matrix $\rightarrow$ computation problems when M is big
- $Var(\boldsymbol{y})^{-1}[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}]$ is computed only one time, for one site we should have computed this operation for each point on the grid
- $Var(\boldsymbol{w}(\boldsymbol{S})|\boldsymbol{y})$ is the prediction uncertainty (only the diagonal, outside we have the covariances)

3.4 Multivariate models

$$\boldsymbol{y}(\boldsymbol{s}) = \boldsymbol{X}(\boldsymbol{s})'\boldsymbol{\beta} + \boldsymbol{\alpha} \odot \boldsymbol{w}(\boldsymbol{s}) + \boldsymbol{\varepsilon}(\boldsymbol{s})$$

$$\boldsymbol{y}(\boldsymbol{s}) = \boldsymbol{X}(\boldsymbol{s})'\boldsymbol{\beta} + \boldsymbol{A}\boldsymbol{w}(\boldsymbol{s}) + \boldsymbol{\varepsilon}(\boldsymbol{s})$$

Remarks:

- $\boldsymbol{y}(\boldsymbol{s})$ is a $p \times 1$ vector, where p are the number of variables we want to model
- $\boldsymbol{w}(\boldsymbol{s}) \sim GP_p\big(\boldsymbol{0}, \boldsymbol{V}\rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta})\big)$ is a p-variate Gaussian random process
- $\boldsymbol{V}$ is a $p \times p$ correlation matrix
- $\boldsymbol{\varepsilon}(\boldsymbol{s}) \sim N_p(\boldsymbol{0}, \boldsymbol{\Sigma}^2_{\varepsilon})$ is a p-variate Normal random variable with $\boldsymbol{\Sigma}^2_{\varepsilon}$ diagonal
- The unknown parameter set is $\Psi = \{\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Sigma}^2_{\varepsilon}, \boldsymbol{\theta}, \boldsymbol{V}\}$

Importante:

- Voglio modellare congiuntamente piú variabili di output nello stesso modello, ossia piú $\boldsymbol{y}$
- Stimiamo un modello unico che modellizza tutte le p variabili e non p modelli univariati
- Conviene scegliere questo modello quando una delle variabili é osservata su pochi siti ma é correlata con le altre, in questo modo sfrutto la correlazione fra le variabili per aggiungere informazione al modello. Nel caso di incorrelazione non mi conviene fare il modello multivariato
- Le nostre covariante nel modello sono dei numeri deterministici senza incertezza, nel caso di incertezza non posso usarla come regressore. Inoltre il regressore dev'essere sempre presente per poter farlo usare come covariata del modello $\rightarrow$ al massimo non uso alcuna covariata
- $\boldsymbol{X}$ é una matrice a blocchi non piú un vettore, e ogni blocco si riferisce a ciascuna variariabile del modello, ossia ciascuna variabile y ha il suo set di covariate che possono differire dalle altre
- $\boldsymbol{\beta}$ anch'esso sará strutturato in sotto-vettori e ciascun sottovettore si riferisce ad una variabile y_i

$$p = 2, \ \textit{covariate di } y_1 \ \textit{e} \ y_2$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} (x_{11} & x_{12}) & 0 & 0 & 0 \\ 0 & 0 & (x_{21} & x_{22} & x_{23}) \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix}$$

- $\boldsymbol{w}(s) \sim GP_p\big(\mathbf{0}, \boldsymbol{V_p}\rho(||\boldsymbol{s}-\boldsymbol{s'}||;\boldsymbol{\theta})\big)$: date p variabili esiste un GP per ciascuna variabile e probabilente diversi tra di loro.
- $\boldsymbol{V}_p \cdot \rho$ matrice dove ogni elemento é una funzione e $\boldsymbol{V}$ é una matrice di correlazione
- $\boldsymbol{\theta}$ parametro che mi gestisce la correlazione spaziale, le p variabili sono obbligate a condividere lo stesso $\boldsymbol{\theta}$ per la funzione di correlazione ρ, questo puó essere un problema perché le variabili posso essere correlate in modo diverso. Tale scelta viene fatta in modo tale che la matrice sia semi-definita positiva $\boldsymbol{V}_p \cdot \rho$
- $\boldsymbol{\varepsilon(s)}$: date p variabili su ciascuna variabile avró un errore di misura differente e con scale differenti, inoltre la varianza risulta essere $\boldsymbol{\Sigma}^2_\varepsilon$ diagonale, perché non ho covarianza tra gli errori, questo perché la correlazione la gestisce $\boldsymbol{w} \rightarrow$ gli errori di m**isura sulle** p **variabili sono indipendeti**
- A é una matrice diagonale $p \times p$

3.5 Why a multivariate model?

- Why not fitting a model for each variable?
- If two or more variables are correlated, spatial prediction can benefit from this correlation
- Especially if one variables is observed at few spatial locations w.r.t. the other variables
- But computing time is higher (matrices are $pn \times pn$ if all variables are observed at n locations)

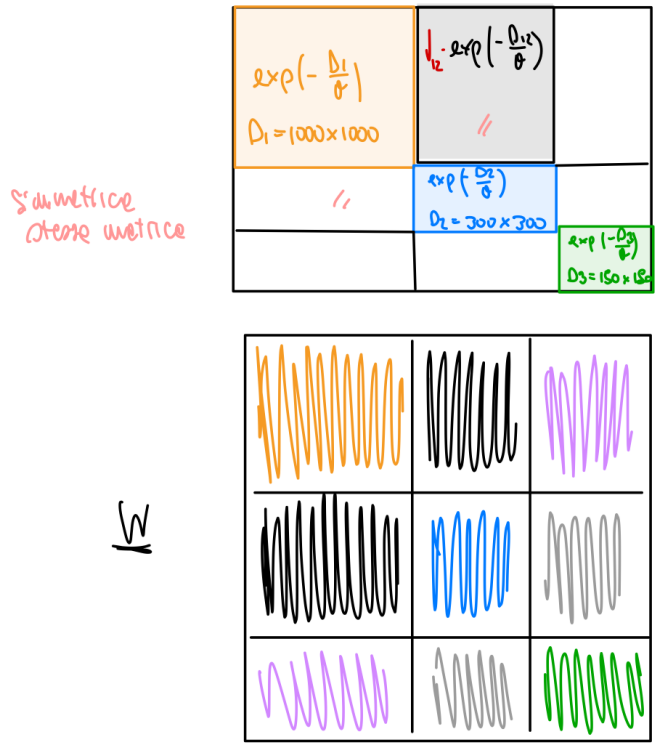
$$\textit{data observed } n \textit{ (sites) } = 1000 \rightarrow 1000 \times 1000 \textit{ matrices}$$

$$\textit{with } p = 5 \rightarrow 5000 \times 5000 \textit{ matrices}$$

Variables observed on n ***sites***

$$n_1 = 1000 \qquad n_2 = 300 \qquad n_3 = 150 \qquad \rightarrow \qquad n = 1450$$

matrice di correlazione del $\boldsymbol{w}$ *dato da un* GP_3



3.6 Bivariate model

- A bivariate model is a (simple) special case of the multivariate model.
- $\boldsymbol{x}_1(\boldsymbol{s})'$ and $\boldsymbol{x}_2(\boldsymbol{s})'$ can have different lengths

$$\begin{bmatrix} y_1(\boldsymbol{s}) \\ y_2(\boldsymbol{s}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1(\boldsymbol{s})' & \mathbf{0} \\ \mathbf{0} & \boldsymbol{x}_2(\boldsymbol{s})' \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_1(\boldsymbol{s}) \\ \boldsymbol{\beta}_2(\boldsymbol{s}) \end{bmatrix} + \begin{bmatrix} \alpha_1 w_1(\boldsymbol{s}) \\ \alpha_2 w_2(\boldsymbol{s}) \end{bmatrix} + \begin{bmatrix} \varepsilon_1(\boldsymbol{s}) \\ \varepsilon_2(\boldsymbol{s}) \end{bmatrix}$$

$$\boldsymbol{V} = \begin{bmatrix} 1 & corr\big(w_1(\boldsymbol{s}), w_2(\boldsymbol{s})\big) \\ corr\big(w_2(\boldsymbol{s}), w_1(\boldsymbol{s})\big) & 1 \end{bmatrix}$$

$$\boldsymbol{\Sigma}_{\varepsilon}^2 = \begin{bmatrix} \sigma_{1,\varepsilon}^2 & 0 \\ 0 & \sigma_{2,\varepsilon}^2 \end{bmatrix}$$

3.7 Linear model of coregionalization

- $\boldsymbol{V}\rho(||\boldsymbol{s}-\boldsymbol{s'}||;\boldsymbol{\theta})\big)$ is called linear coregionalization model
- $\boldsymbol{\theta}$ is common to the p variables is really difficult to identify
- This could be a limit because all the variables are forced to share the same spatial correlation (same function and strenght)
- $\boldsymbol{V}$ is a symmetric correlation matrix (diagonal elements equal to 1), only $\frac{p(p-1)}{2}$ elements of $\boldsymbol{V}$ are estimated

3.8 Data structure

A multivatiate data set can be classified depending on spatial locations:

- **Isotopic:** all p variables are observed at the same n spatial locations.
  - There is a single distance matrix, as all variables share the same set of spatial locations.
  - $\boldsymbol{\alpha}\odot\boldsymbol{w(s)}$ I can directly execute the alpha operation inside $\boldsymbol{w(s)}$ and I will have not a correlation matrix but a covariance one. Estimated in closed form
- **Fully heterotipic:** the p variables do not share a single spatial location.
  - There are p distance matrices, each corresponding to a different variable.
  - $\boldsymbol{\alpha}\odot\boldsymbol{w(s)}$ can't be estimated in closed form because the $\boldsymbol{w}$ matrix is rectangular
- **Partially heterotopic:** : some of the p variables are observed at a subset of the n spatial locations. This is the most common case and it is the case handled by D-STEM.
  - There are p distance matrices, each corresponding to a different variable. However, the matrices may have different dimensions based on the subset of spatial locations where each variable is observed.
  - $\boldsymbol{\alpha}\odot\boldsymbol{w(s)}$ can't be estimated in closed form because the w matrix is rectangular

4 Lesson 4

4.1 Spatio-temporal model: the dynamic coregionalization model (DCM)

$$y(\boldsymbol{s},t) = \boldsymbol{x_\beta}(\boldsymbol{s},t)'\boldsymbol{\beta} + \boldsymbol{x_z}(\boldsymbol{s})'\boldsymbol{z}(t) + \alpha w(\boldsymbol{s},t) + \varepsilon(\boldsymbol{s},t)$$
$$\boldsymbol{z}(t) = \boldsymbol{G}\boldsymbol{z}(t-1) + \boldsymbol{\eta}(t)$$

Remarks:

- $\boldsymbol{x_\beta}(\boldsymbol{s},t)$ and $\boldsymbol{x_z}(\boldsymbol{s})$ are vectors of covariates, they cant contain missing values
- $w(\boldsymbol{s},t) \sim GP\big(0, \rho(||\boldsymbol{s}-\boldsymbol{s}'||;\boldsymbol{\theta})\big)$ is correlated over space but IID over time
- $\boldsymbol{z}(t)$ is $q \times 1$ dimensional with Markovian dynamics
- $\boldsymbol{G}$ is a stable $q \times q \Rightarrow$ transition matrix, it can be full, diagonal
- $\boldsymbol{\eta}(t) \sim N(\boldsymbol{0}, \Sigma_\eta)$ is the innovation with Σ_η the variance-covariance matrix
- $\varepsilon(\boldsymbol{s},t) \sim N(0, \sigma^2_\varepsilon)$ is the measurement error, time invariant (constant across time)
- **Example:** we have only one covariate altitude which is time invariant and $y(\boldsymbol{s},t)$ is the pollutant concentration. The global effect of the altitude is placed in $\boldsymbol{x_\beta}(\boldsymbol{s},t)$ which is constant across space and time but we have a **temporal correction** given by $\boldsymbol{x_z}(\boldsymbol{s})'\boldsymbol{z}(t)$. This because the effect of the altitude is different in winter and in summer. So if we put this covariante in both $\boldsymbol{x}'s$ the $\boldsymbol{x_z}(\boldsymbol{s})'\boldsymbol{z}(t)$ is an **correction/adjustment** in respect of the global term $\boldsymbol{x_\beta}(\boldsymbol{s},t)$. We can choose in our model where to place this covariate maybe just in $\boldsymbol{x_z}(\boldsymbol{s})'\boldsymbol{z}(t)$ anyway we can do every combination we want. Best way to choose where to place the covariantes is by → **cross-validation** and pick up the best model.

Insights:

- $y(\boldsymbol{s},t)$, where $\boldsymbol{s}$ is a vector that indicates the spatial location such as $(latitude, longitude)$ and t is the temporal dimension that will be discrete
- $\boldsymbol{z}(t)$ latent variable which is only temporal that means that is constant across space
- $\boldsymbol{x_z}(\boldsymbol{s})$ is a covariate vector, if we want that $\boldsymbol{z}(t)$ interacts with some covariates, these covariates must be spatial (means that those dont change across time such as elevation while wind is spatial and temporal covariate). If the covariates are spatial and teporal we lose **model identifiability** or in oder word we are **overfitting**.
- $\alpha w(\boldsymbol{s},t) \sim GP$ and we have a GP for each time and they are indipendent → **assumptions/forcing**. Also we have a different θ for each which can be $\theta(t)$, but since this parameter is not very well indentifiable we can fix this parameter
- **Markovian dynamics** refers to the temporal evolution of the latent variable $\boldsymbol{z}(t)$. Specifically, it means that the state of the system at time t (represented by $\boldsymbol{z}(t)$) depends only on the state at the previous time step, $t-1$, and not on any earlier time steps. So the current state retains information from the immediate past but is not influenced by events or states further back in time.
- $\boldsymbol{G}$ is a stable matrix which means that $\boldsymbol{z}(t)$ doesnt diverge when time increses
- $\varepsilon(\boldsymbol{s},t)$ scalar because $y(\boldsymbol{s},t)$ is scalar

4.2 Why the Markovian dynamics?

- The Markovian dynamic helps to describe the temporal persistence which usually characterizes temporal phenomena.
- For instance: even if we stop air pollution emissions, pollutant concentration will not drop instantly to zero. ⇒ Emission would be a model covariate
- So using only the covariantes cant describe this dynamics therebefore we model the **persistency** using $\boldsymbol{z}(t)$. Even if we use one spatial model everyday we cant caputure this effect.

4.3 Data matrix

- $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T)$ is the data matrix
- Each $\boldsymbol{y}_t$ is the vector of spatial observation at time $t = 1, \ldots, T$
- In each $\boldsymbol{y}_t$, missing data are possible
- Covariates are in a data array $\boldsymbol{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_T\}$, each $\boldsymbol{X}_t$ is a $n \times b$ matrix where b is the number of covariates. $\boldsymbol{X}$ cannot have missing data. One way to avoid missing data is throught **interpolation/not kriging**
- So every t, based on the frequency of our dataset which is constant, we have a vector of observe quantity.
- Spatial locations $\boldsymbol{s}$ can be fixed in our dataset but its common that maybe stations can be broken in some days and record from other stations. So generally in a more general scenario spatial locations $\boldsymbol{s}$ can change. (e.g. smarthphone network)

4.4 Likelihood function

$$L(\Psi; \boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{X}) = L(\Psi; \boldsymbol{Y} | \boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{X}) \cdot L(\Psi; \boldsymbol{Z}) \cdot L(\Psi; \boldsymbol{W})$$

- The parameter vector is $\Psi = \{\boldsymbol{\beta}, \alpha, \sigma^2_\varepsilon, \boldsymbol{\theta}, \boldsymbol{G}, \boldsymbol{\Sigma}_\eta\}$
- $\boldsymbol{W}$ and $\boldsymbol{Z}$ have different dimensions. $\boldsymbol{w}$ has the dimension of the network instead the dimension of $\boldsymbol{z}$ is based on the number of covariantes
- $\boldsymbol{W} = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T\}$ for each time step we have a vector $n \times 1 \Rightarrow$ matrix. In the model w is a scalar because we are explaining the output at spatial location $\boldsymbol{s}$, but we have n spatial locations $(n \times 1)$
- $\boldsymbol{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_T\}$ for each time step we have a vector $q \times 1 \Rightarrow$ matrix
- $\boldsymbol{X}$ are covariates
- We can do the factorization of the Likelihood above because $\boldsymbol{W} \perp \boldsymbol{Z} \rightarrow$ **model assumption**. And also in this factorization We are supposing to know $\boldsymbol{W}$ and $\boldsymbol{Z}$ but in practise we dont.

4.5 Log-likelihood function

$$\begin{aligned} -2log\big(L_\Psi\big) = & \; T log|\boldsymbol{\Sigma}_\varepsilon| + \sum_{t=1}^{T} \boldsymbol{e}_t' \boldsymbol{\Sigma}_\varepsilon^{-1} \boldsymbol{e}_t \\ & + T log|\boldsymbol{\Sigma}_\eta| + \sum_{t=1}^{T} (\boldsymbol{z}_t - \boldsymbol{G}\boldsymbol{z}_{t-1})' \boldsymbol{\Sigma}_\eta^{-1} (\boldsymbol{z}_t - \boldsymbol{G}\boldsymbol{z}_{t-1}) \\ & + T log|\boldsymbol{\Sigma}_w| + \sum_{t=1}^{T} \boldsymbol{w}_t' \boldsymbol{\Sigma}_w^{-1} \boldsymbol{w}_t \end{aligned}$$

Innovation:

$$\boldsymbol{z}(t) = \boldsymbol{G}\boldsymbol{z}(t-1) + \boldsymbol{\eta}(t)$$

$$\boldsymbol{\eta}(t) = \boldsymbol{z}(t) - \boldsymbol{G}\boldsymbol{z}(t-1)$$

Where:

- $\boldsymbol{e}_t = \boldsymbol{y}_t - \boldsymbol{X}_{\beta,t}\boldsymbol{\beta} - \boldsymbol{X}_{z,t}\boldsymbol{z}_t - \alpha \boldsymbol{w}_t$
- $\boldsymbol{\Sigma}_\varepsilon = \sigma^2_\varepsilon \boldsymbol{I}_n$, with $\boldsymbol{I}_n$ the identity matrix of dimension n
- $\boldsymbol{\Sigma}_w$ is the $n \times n$ correlation matrix (e.g. $exp(-D/\theta)$, with D the distance matrix)
- Maximizing the log likelihood means find the $\boldsymbol{G}$ and the $\boldsymbol{\Sigma}_\eta$ which explains the innovation that we are observing $\Rightarrow$ but we are not observing $\boldsymbol{z}$ such as $\boldsymbol{w}$

4.6 Model estimation – EM algorithm

- Model estimation is (again) based on the EM algorithm
- $E\big(w(\boldsymbol{s},t)|\boldsymbol{Y}\big)$ and $Var\big(w(\boldsymbol{s},t)|\boldsymbol{Y}\big)$ are given by the same formulas of $E\big(w(\boldsymbol{s})|\boldsymbol{Y}\big)$ and $Var\big(w(\boldsymbol{s})|\boldsymbol{Y}\big)$ (because $w(\boldsymbol{s},t)$ are IID over time)
- $E\big(\boldsymbol{z}(t)|\boldsymbol{Y}\big)$ and $Var\big(\boldsymbol{z}(t)|\boldsymbol{Y}\big)$ are given by the Kalman smoother
- However, $Cov\big(\boldsymbol{z}(t), w(\boldsymbol{s},t)|\boldsymbol{Y}\big) \neq \boldsymbol{0}$. E-step and M-step are more complicated than the spatial case.

4.7 DCM alternative formulas

$$y(\boldsymbol{s},t) = \boldsymbol{x_\beta}(\boldsymbol{s},t)'\boldsymbol{\beta} + \boldsymbol{x_z}(\boldsymbol{s})'\boldsymbol{z}(t) + w(\boldsymbol{s},t) + \varepsilon(\boldsymbol{s},t)$$
$$\boldsymbol{z}(t) = \boldsymbol{G}\boldsymbol{z}(t-1) + \boldsymbol{\eta}(t)$$
$$w(\boldsymbol{s},t) = \sum_{j=1}^{c} a_j x_j(\boldsymbol{s},t) w_j(\boldsymbol{s},t)$$

$$y(\boldsymbol{s},t) = \mu(\boldsymbol{s},t) + w(\boldsymbol{s},t) + \varepsilon(\boldsymbol{s},t)$$

$\mu(\boldsymbol{s},t) = x_\beta(\boldsymbol{s},t)\boldsymbol{\beta}$	*fixed effect model*
$w(\boldsymbol{s},t) = \sum_{j=1}^{c} a_j x_j(\boldsymbol{s},t) w_j(\boldsymbol{s},t) + x_z(\boldsymbol{s},t)\boldsymbol{z}(t)$	*random effect model*
$\boldsymbol{z}(t) = \boldsymbol{G}\boldsymbol{z}(t-1) + \boldsymbol{\eta}(t)$	*Markovian dynamics*

5 Lesson 5

5.1 Spatio-temporal model: the (univariate) hidden dynamic geostatistical model (HDGM)

$$y(\boldsymbol{s},t) = \boldsymbol{x}_{\boldsymbol{\beta}}(\boldsymbol{s},t)'\boldsymbol{\beta} + \alpha z(\boldsymbol{s},t) + \varepsilon(\boldsymbol{s},t)$$
$$z(\boldsymbol{s},t) = gz(\boldsymbol{s},t-1) + \eta(\boldsymbol{s},t)$$

Remarks:

- $\eta(\boldsymbol{s},t) \sim GP\big(0, \rho(||\boldsymbol{s}-\boldsymbol{s}'||;\boldsymbol{\theta})\big)$ is correlated over space but IID over time
- $z(\boldsymbol{s},t)$ is scalar and has Markovian dynamic
- α is a scale coefficient (v in D-STEM)
- g is the transition coefficient
- $\varepsilon(\boldsymbol{s},t) \sim N(0,\sigma^2_\varepsilon)$ is the measurement error
- The model parameter set is $\Psi = \{\boldsymbol{\beta}, \alpha, \sigma^2_\varepsilon, \boldsymbol{\theta}, g\}$

5.2 Spatio-temporal model: the (multivariate) hidden dynamic geostatistical model (HDGM)

$$\boldsymbol{y}(\boldsymbol{s},t) = \boldsymbol{X}_{\boldsymbol{\beta}}(\boldsymbol{s},t)'\boldsymbol{\beta} + \boldsymbol{z}(\boldsymbol{s},t) + \boldsymbol{\varepsilon}(\boldsymbol{s},t)$$
$$\boldsymbol{z}(\boldsymbol{s},t) = \boldsymbol{G}\boldsymbol{z}(\boldsymbol{s},t-1) + \boldsymbol{\eta}(\boldsymbol{s},t)$$

Remarks:

- $\boldsymbol{y}(\boldsymbol{s},t)$ and $\boldsymbol{z}(\boldsymbol{s},t)$ are $p \times 1$ vectors
- $\eta(\boldsymbol{s},t) \sim GP_p\big(\boldsymbol{0}, \boldsymbol{V}\rho(||\boldsymbol{s}-\boldsymbol{s}'||;\boldsymbol{\theta})\big)$ is a p-variate Gaussian random process
- $\boldsymbol{V}$ is a variance-covariance matrix (in Calculli et al. $\boldsymbol{V}$ is a correlation matrix and there is the scaling matrix $\boldsymbol{A}$)
- $\boldsymbol{z}(\boldsymbol{s},t)$ has Markovian dynamic
- $\boldsymbol{G}$ is a diagonal $p \times p$ transition matrix
- $\boldsymbol{\varepsilon}(\boldsymbol{s},t) \sim N_p\big(\boldsymbol{0}, \boldsymbol{\Sigma}_\varepsilon\big)$ is a p-variate Normal random variable with $\boldsymbol{\Sigma}_\varepsilon$ diagonal
- The model parameter set is $\Psi = \{\boldsymbol{\beta}, \boldsymbol{\Sigma}_\varepsilon, \boldsymbol{\theta}, \boldsymbol{V}, \boldsymbol{G}\}$

Insights and differences in respect to DCM:

- we have only 1 lantent variable which is $\boldsymbol{z}(\boldsymbol{s},t)$ and this a map.
- With the DCM you tend to overfit for each time step u are able to fit something which is spatial and which is indipedent from the previous time step and u have no constraints. U can change $\boldsymbol{w}(\boldsymbol{s},t)$ at the previous time step.
- DCM has high R^2 but it doesnt perform really well in cross validation because u are overfitting

5.3 Model estimation

- The HDGM is estimated similarly to the DCM
- But we only have the $z(\boldsymbol{s},t)$ latent variable which is estimated in the E-step by the Kalman smoother.
- Spatial prediction is also done by the Kalman smoother assuming that y is not observed at the spatial prediction locations (it is added as NaN in the y vector).

6 Lesson 6

6.1 Towards spatio-temporal functional models

Remarks:

- In many cases, data are observed at high frequency/resolution in at least one dimension (spatial or temporal). It usually happens with the temporal dimension, for instance, pollutant concentrations observed hourly or every 15 minutes.
- In general, high frequency observations (100, 1000, 10.000 per day)
- In space it is less common because a high resolution sampling is usually very expensive. In 3D space, one dimension may be sampled at higher resolution than the others

Disadvantages:

- Usually the original data set is very large and so the computational burden
- Data may be collected asynchronously over time (e.g., different monitoring stations may have different clocks)
- Data may have large gaps over time (how does the Kalman Smoother perform in this case?)
- Temporal correlation is usually very high (the Markovian model may explain the data but it is not very useful for prediction)

6.2 Functional data analysis (FDA)

- In FDA, the object of the statistical inference is a continuous function rather than scalar/vector values
- For instance, the temperature measured by a sensor over the 24h of the day can be described by a (smooth?) function:
  - Independently of how many observations we take
  - Independently of where in time these observations are taken
- Which function or class of functions should we use?
  - The function should describe the global data pattern
  - In a way, the function filters out the data noisy
  - The researcher should be able to control the function smoothness

6.3 Basis Functions

We are going to choose B-spline bases as it is the most flexible basis to describe complex functions shape. We could also use other kind of basis, but one thing to keep in mind is that the number of basis function must be fixed before model estimation. High p (number of basis) usally means better R^2, but overfitting may be an issue. Usually when using an FDA approach it is recommended to use a high p, but in the case of f-HDGM this is not viable since the covariance matrices involved have dimension $(n * p) \times (n * p)$. Since n is usually large, a high p would make the estimation unfeasible.

Remarks:

- Spline is a class of functions which allows to easily control the function smoothness:
  - by selecting the proper basis functions
  - by selecting the proper knots
- B-spline basis are useful to describe non-periodic functions. Knots can be placed ad-hoc along the function domain (more knots where the function should change more rapidly)
- Fourier basis can describe periodic functions. Smoothness is controlled by the number of basis

6.4 How to describe functional data in a space-time model?

- We now want to model the generic observation $y(\boldsymbol{s}, t, h)$ where:
  - $\boldsymbol{s}$ and t are the usual spatial and temporal indexes
  - $h \in \mathbb{R}$ is the functional dimension (spatial or temporal)
- Examples:
  - h could describe the continuous time within the day while t is the index of days
  - h could describe altitude in a 3D space while s describes the generic location across the globe

6.5 The functional HDG model in D-STEM

$$y(\boldsymbol{s}, t, h) = f(\boldsymbol{s}, t, h) + \varepsilon(\boldsymbol{s}, t, h)$$
$$f(\boldsymbol{s}, t, h) = x(\boldsymbol{s}, t, h)'\boldsymbol{\beta}(h) + \boldsymbol{\phi}(h)'z(\boldsymbol{s}, t)$$
$$z(\boldsymbol{s}, t) = \boldsymbol{G}\boldsymbol{z}(\boldsymbol{s}, t-1) + \eta(\boldsymbol{s}, t)$$

Remarks:

- $\eta(\boldsymbol{s}, t) \sim GP\big(0, \rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta})\big)$ is correlated over space but IID over time.
- $z(\boldsymbol{s}, t)$ is not scalar anymore and has Markovian dynamic
- G is a diagonal transition matrix with diagonal elements in the $p \times 1$ vector g
- $\varepsilon(\boldsymbol{s}, t) \sim N(0, \sigma^2_\varepsilon)(h)$ is a Guassian measurement error independent in space and time. This means that the variance changes for each point of h, but it isn't influenced by position or time.
  $\boldsymbol{\phi}(h)$ is a $p \times 1$ vector of basis functions evaluated at h, while $\boldsymbol{c}_\varepsilon$ is a vector of coefficients to be estimated.
  $$log(\sigma^2_\varepsilon(h)) = \boldsymbol{\phi}(h)'\boldsymbol{c}_\varepsilon$$
- The model parameter set is $\Psi = (\boldsymbol{c_\varepsilon}', \boldsymbol{c_\beta}', \boldsymbol{g}', \boldsymbol{v}', \boldsymbol{\theta}')'$

Insights:

- Since ε depends on parameter h this means that we are assuming ε to be heteroskedastic across the domain of the h variable. This means that for each value of h we have a different value of ε
- The overall idea now is to have basis function and find the optimal coefficients to

7 Approfondimenti

$$\hat{w}(\boldsymbol{s}) \in \mathbb{R},\ \textit{latent variable} \qquad\qquad w(\boldsymbol{s}) \sim GP\big(0, \rho(||\boldsymbol{s} - \boldsymbol{s}'||; \boldsymbol{\theta})\big)$$

7.1 Kriging

kriging: is a spatial interpolation and prediction technique used in geostatistics to estimate values at unobserved locations based on the values observed at nearby locations.

- kriging also known as Gaussian process regression, is a method of interpolation based on Gaussian process governed by prior covariances
- Under suitable assumptions of the prior, kriging gives the best linear unbiased prediction (BLUP) at unsampled locations
- it provides not only point predictions but also estimates of the prediction uncertainty.
- kriging can be used to predict the values of the latent variable, w(s), at unsampled spatial locations. This is particularly useful when you want to estimate pollutant concentrations at locations not covered by monitoring stations.
- Kriging predicts the value of a function at a given point by computing a weighted average of the known values of the function in the neighborhood of the point. The method is closely related to regression analysis. Both theories derive a best linear unbiased estimator based on assumptions on covariances

Steps:

- The first step in spatiotemporal kriging is to estimate the spatiotemporal variogram. This involves calculating the variance of the differences in values at different spatial locations and time points, considering both spatial and temporal distances. The variogram is used to model the spatiotemporal correlation structure of the data.
- The variogram is related to the covariance function, which represents the strength of the correlation between observations at different locations and times. The covariance function is used to construct a covariance matrix that accounts for both spatial and temporal dependencies.
- A kriging system is set up based on the covariance matrix, and the system includes observations at both observed and unobserved locations and times. The kriging system is then solved to obtain kriging weights.
- The kriging weights obtained from the system are used to predict values at unobserved spatial locations and time points. The kriging prediction is a linear combination of observed values with weights determined by the kriging system.
- Similar to traditional kriging, spatiotemporal kriging provides estimates of the prediction uncertainty. The kriging variance at each unobserved location and time point reflects the uncertainty associated with the predictions.
- The performance of the spatiotemporal kriging model can be assessed through cross-validation or other validation techniques to ensure its reliability and generalizability.

Key points related to kriging:

- **Variogram:** Kriging relies on the variogram, which quantifies the spatial correlation or covariance between data points at different spatial lags. The variogram model is used to describe how the spatial correlation decreases as the distance between locations increases.
- **Spatial Weights:** Kriging assigns spatial weights to observed data points, and these weights are used to estimate values at unobserved locations. The weights depend on the spatial correlation structure defined by the variogram.

7.2 Gaussian process

- a Gaussian process is a **stochastic process** (a collection of random variables indexed by time or space), such that every finite collection of those random variables has a multivariate normal distribution, i.e. every finite linear combination of them is normally distributed.
- The distribution of a Gaussian process is the joint distribution of all those (infinitely many) random variables, and as such, it is a distribution over functions with a continuous domain (time or space)
- Gaussian processes can be seen as an infinite-dimensional generalization of multivariate normal distributions. Gaussian $\sim$ Normal
- A Gaussian stochastic process is strongly stationary or strict-sense stationary if and only if it is weak-sense stationarity, wide-sense stationarity (WSS).
- Basic aspects that can be defined through the covariance function are the process' stationarity, isotropy, smoothness and periodicity
  - If the process is stationary, the covariance function depends only on $\boldsymbol{x} - \boldsymbol{x'}$
  - If the process depends only on $|\boldsymbol{x} - \boldsymbol{x'}|$ (E. distance) then the process is considered isotropic.
  - A process that is concurrently stationary and isotropic is considered to be homogeneous
- Gaussian processes translate as taking priors on functions and the smoothness of these priors can be induced by the covariance function. If we expect that for "near-by" input points $\boldsymbol{x}$ and $\boldsymbol{x'}$ their corresponding output points $\boldsymbol{y}$ and $\boldsymbol{y'}$ then the assumption of continuity is present. If we wish to allow for significant displacement then we might choose a rougher covariance function.

Conclusions:

- $w(\boldsymbol{s_i})$ is a random variable and given a specific $\boldsymbol{s_i}$, the value of $w(\boldsymbol{s_i}) \in \mathbb{R}$ comes from a distribution
- A GP is continuous in space. This means that:
  - It can be simulated on a regular grid or
  - On a irregular grid
  - For any given number of spatial locations
- We impose that $w(\boldsymbol{s}) \sim GP$, and then im going to take a realization of a gaussian process and use it as my latent variable per each point in my dataset. I don't have any troubles because the GP is countinous and is definited everywhere
- When we talk about $w(\boldsymbol{s})$ we are referring to the whole realization so the value of each $w(\boldsymbol{s})_i$

Insights:

- a stationary process is a stochastic process whose unconditional joint probability distribution does not change when shifted in time. Consequently, parameters such as mean and variance also do not change over time.
- Since stationarity is an assumption underlying many statistical procedures used in time series analysis, non-stationary data are often transformed to become stationary. The most common cause of violation of stationarity is a trend in the mean, which can be due either to the presence of a unit root or of a deterministic trend.
- A trend stationary process is not strictly stationary, but can easily be transformed into a stationary process by removing the underlying trend, which is solely a function of time. Similarly, processes with one or more unit roots can be made stationary through differencing. An important type of non-stationary process that does not include a trend-like behavior is a cyclostationary process, which is a stochastic process that varies cyclically with time.

7.3 Apulia paper

- We applied the HDGM to model two pollutants + include in the same model some meteorological variables **observed in another network** also as output of the model
- So we have a model with 8 variables where 2 are pollutants and 6 meteorological variables
- Why didnt u use the meteorological variables as covariates ? because the grids are different, so one station might measure the PM10 but not all the meteorological variables. If I would like to use the meteorological variables as covariates I have to interpolate the covariates using any alogorithm for spatial interpolation, but the problem is everytime u interpolate u have the number of the interpolation but u dont have the uncertenty. Another thing is that we could have missing values measuring the meteorological variables → and they are not allowed.
- For us the covariates have 0 uncertenty because are just numbers
- the $\boldsymbol{V}$ correlation matrix, considers the cross-correlation between the all 8 variables
- The estimated model once we ran the EM was to use it in order to do spatial interpolation
- We basically used the information from the meteorological station to predict the pollutants concentration in the area where the station is located
- **The problem** is that when we consider in the model pollutants + meteorological variables we **lose the casuality** of the effect that the weather has on the pollutants concentration, usually its the weather that affects the pollutants concentration and not the opposite but our model doesnt know this. So this is the limit of this model, some outputs may not make sense.

7.4 Project clarifications

- **Standardizzare** significa trasformare una distribuzione di dati in modo che abbia una media di zero e una deviazione standard di uno. Questo processo è comunemente utilizzato nelle analisi statistiche e nei modelli di machine learning per rendere più facile la comparazione tra variabili che potrebbero avere scale di misura diverse o per facilitare l'interpretazione dei risultati.

  La standardizzazione di una variabile X può essere espressa matematicamente attraverso la seguente formula:

$$X \sim d(\mu, \sigma^2) \quad \rightarrow \quad Z \sim d(0,1)$$

$$Z = \frac{X - \mu}{\sigma}, \qquad -1 \leqslant Z \leqslant 1$$

X	*è il valore originale della variabile*
Z	*è il valore standardizzato della variabile X*
μ	*è la media della distribuzione dei dati*
σ	*è la deviazione standard della distribuzione dei dati*

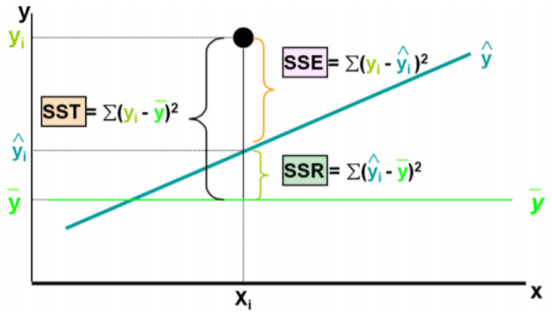
- La radice dell'errore quadratico medio **(RMSE)** è una misura della discrepanza tra i valori osservati e i valori predetti da un modello statistico o di previsione. È comunemente utilizzato per valutare la bontà di adattamento di un modello

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

  Standardizzando, la mia response variable é compresa tra -1 e 1, dunque gli errori che commetto sono in questa scala. I residui sono poi elevati al quadrato e messi sotto radice, posso dunque vedere questo come un errore in percentuale che commetto.

- La **normalizzazione** é che è un tipo di standardizzazione in cui i dati vengono trasformati in modo che cadano entro un intervallo specifico, spesso tra -1 e 1 o tra 0 e 1
- **Il coefficiente di determinazione**, comunemente indicato come R^2 (R al quadrato), è una misura statistica che indica quanto bene un modello si adatta ai dati osservati. In altre parole, R^2 misura la proporzione di varianza della variabile dipendente che viene spiegata dalle variabili indipendenti nel modello.

Sum of squares total	$SST = \sum_{i=1}^{n}(y_i - \overline{y})^2 = SSR + SSE$
Sum of squares Error	$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}\varepsilon_i^2$
Sum of squares due to regression (SSR)	$SSE = \sum_{i=1}^{n}(\hat{y}_i - \overline{y})^2$
R squared	$R^2 = 1 - \frac{SSE}{SST}$



Se ottieni un coefficiente di determinazione R^2 **negativo**, ciò indica che il tuo modello di regressione è peggiore del semplice modello medio. In altre parole, il modello non spiega alcuna variazione o addirittura peggiora la capacità di predizione rispetto a una previsione basata semplicemente sulla media della variabile dipendente.

- **LBQ Test (Ljung-Box Q-test):** Il test LBQ viene utilizzato per verificare l'autocorrelazione residua nei residui di un modello di serie storica. **L'ipotesi nulla** del test LBQ è che non ci sia autocorrelazione nei residui. Se il risultato del test è 0, ciò indica che non vi è autocorrelazione residua nei residui del modello. Se il risultato è diverso da 0, indica che vi è autocorrelazione residua nei residui del modello.

H0: *no autocorrelazione tra residui*

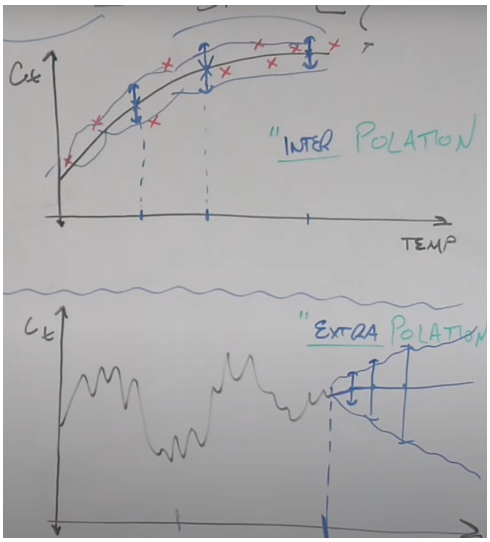
- **ARCH Test (Autoregressive Conditional Heteroskedasticity):** Il test ARCH viene utilizzato per verificare la presenza di eteroschedasticità condizionale nei residui di un modello di serie storica. **L'ipotesi nulla** del test ARCH è che non ci sia eteroschedasticità condizionale nei residui. Se il risultato del test è 0, ciò indica che non vi è eteroschedasticità condizionale nei residui del modello. Se il risultato è diverso da 0, indica che vi è eteroschedasticità condizionale nei residui del modello.

H0: *residui omoschedastici*

8 Time series analysis

8.1 Introduction:

- **Interpolation:** predictiong from within a range of observable data. Based on the number of observable data u have u are going to build a more accurate model and residuals that are retrived from the model are pretty much the same in terms of value for each data point. This happens because the uncertainty is the same within the data and becomes higher when we are going outside from the observable data → non time series problems are related to interpolation problems.
- **Extrapolation:** the further we go from our data points the higher the uncertaincy is on our prediction. If Im making a two days prediction in the future, the uncertaincy is going to propagate (sommable), so the uncertaincy of the prediction is going to be added to the uncertaincy of yesterday → Time series problems are related to extrapolation problems.



8.2 Autocorrelation (ACF) and partialcorrelation (PACF) function:

- **ACF:** tell us the correlation the effect at ago lags and the actual one, taking into to account the direct and indirect correalation effect
- **PACF:** we only care of the direct effect, we don't care of the effect of indirect correlations

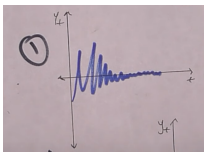
$$\underset{s(t-2)}{J} \rightarrow \underset{s(t-1)}{J} \rightarrow \underset{s(t)}{J}$$

$$corr(J, M) = \textit{direct correlation effect } J \rightarrow M$$

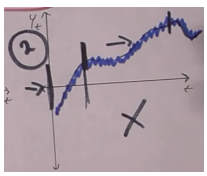
$$corr(J, M) = \textit{indirect correlation effect } J \rightarrow F \rightarrow M$$

8.3 Stationarity

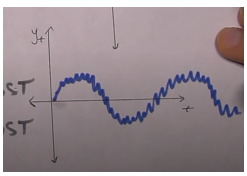
- Most of the models assume that the time series that we are trying to use them on is stationary
- **Def:** a time series is stationary when: μ is constant over time, σ is constant over time and there is no seasonality
- Seasonality periodic behaviour over time that is predictable



μ is constant
no seasonality
σ is not constant



μ is not constant (shifting over time)
no seasonality
σ is constant (no fluctuations)



μ is constant
seasonality (perodic component)
σ is constant

- white noise signal is stationary by definition
- how to check if the process is stationary: visually, global versus local tests (divide the process into chunks), statistical tests such as "Augmented dickey-fuller (ADF test)"
- How to make a time siries stationary:

y(t) is not stationary

$$y_t = \beta_0 + \beta_1 t + \varepsilon_t$$

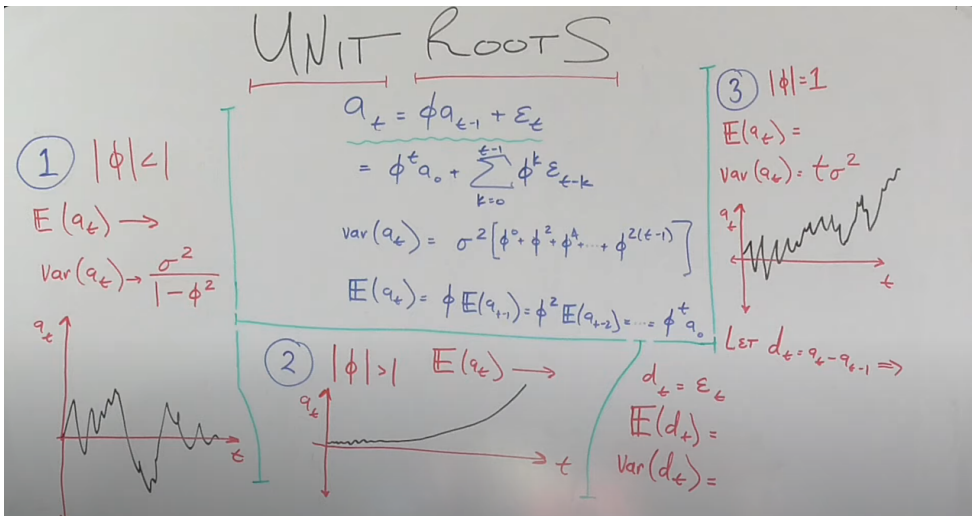
$$\begin{aligned} z_t &= y_t - y_{t-1} \\ &= (\beta_0 + \beta_1 t + \varepsilon_t) - (\beta_0 + \beta_1 (t-1) + \varepsilon_{t-1}) \\ &= \beta_1 + \varepsilon_t - \varepsilon_{t-1} \end{aligned}$$

$$\mathbb{E}[z_t] = \beta_1 (constant)$$

$$var[z_t] = 0 + k^2 + k^2 = 2k^2 (constant)$$

8.4 Unit Roots

- If we have a time series with a unit root then the series is not stationary, so we have to apply some transformation in order to get it stationary
- We have unit root in example 3: $|\phi| = 1$
- $AR(1)$ it can be written as a $MA(\infty)$, try to write recursively $AR(1)$ starting from a a_t
- the first value of the time series: a_0



Example 1: $|\phi| < 1$

$$\phi = 0.5$$

$$\mathbb{E}[a_t] = 0 \rightarrow \phi^\infty = 0$$

$$var[a_t] = \frac{\sigma^2}{1 - \phi^2}$$

Stationary ts

Example 2: $|\phi| > 1$

$$\phi = 2$$

$$\mathbb{E}[a_t] = 0 \rightarrow \phi^\infty = \pm\infty$$

Non stationary ts

Example 3: $|\phi| = 1$

$$\mathbb{E}[a_t] = a_0$$

$$var[a_t] = t \cdot \sigma^2$$

Non stationary ts but ***first difference***

First difference

new series: $d_t = a_t - a_{t-1} = \varepsilon_t$

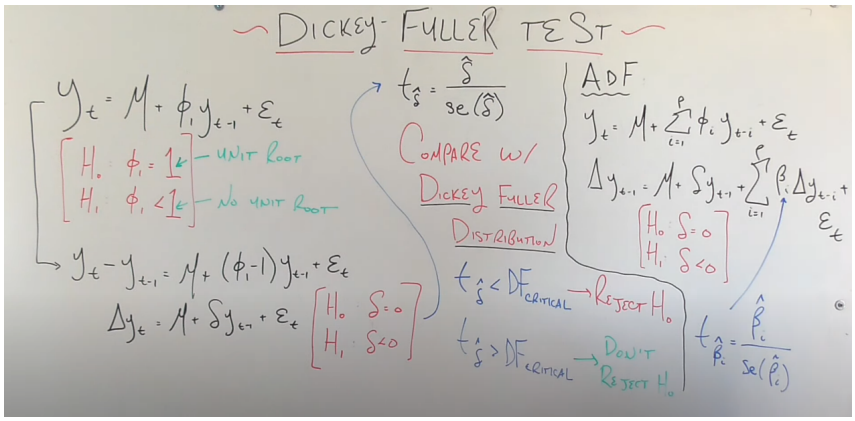
$$\mathbb{E}[d_t] = 0$$

$$var[d_t] = \sigma^2$$

Now is stationary

8.5 Dickey Fuller Test and Augmented Dickey Fuller Test

- Test in order to check if the time series is stationary (mean, variance are constant over time and no seasonality)
- Dickey Fuller Test assumes that our time series in question is a $AR(1)$ process
- *se:* standard error
- We compare the t-statistic against the dickey fuller distribution and not the normal distribution
- We extend this process to a more complex models than $AR(1)$ and thats how **Augmented Dickey Fuller Test** comes into play. We also can calculate the t-distribution for each β_i and we can see if those β_i are significative or not



8.6 White noise

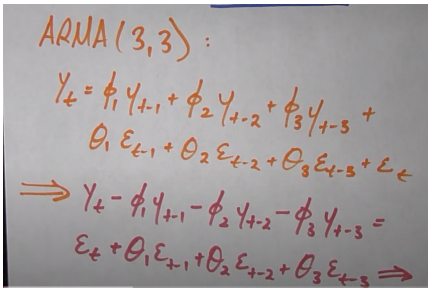
- white noise is a type of time series which has:

mean $= 0$	*st. dev is constant with time*
corr. between lags is 0	*WN is not predictable*

- Whenever we are doing a ts analysis on some y_t we always assume that $y_t = signal + noise$, where I can predict the *signal* and *noise* $\sim WN$, so unpredictable. If we are doing the things right we will end up with WN residuals and we can tell for sure out estimated model is really good.
- To test this we can: visual tests, check ACF and global vs local checks have to match (window that I shift every time and I calculate mean and variance and those have to be the same each time).
- The ACF checks for correlation between lags, if we see anything outside the bands we have a correaltion in that point

8.7 Backshift operator: lag operator

- Write in a more compact way the series
- we are going to trasform the first equation
- we bring all the AR terms on one side and the MA on the other one
- **lag operator:** L or B



$$Ly_t = y_{t-1}, \quad L^2 y_t = y_{t-2}$$

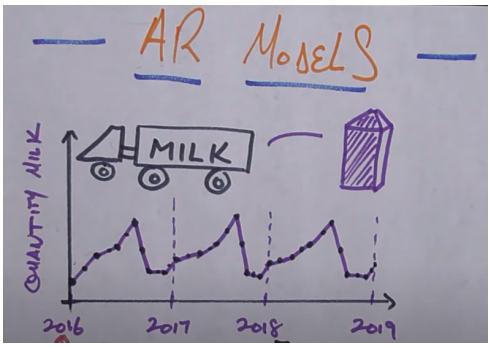
$$y_t - \phi_1 L y_t - \phi_2 L^2 y_t - \phi_3 L^3 y_t = \varepsilon_t - \theta_1 L\varepsilon_t - \theta_2 L^2 \varepsilon_t - \theta_3 L^3 \varepsilon_t$$

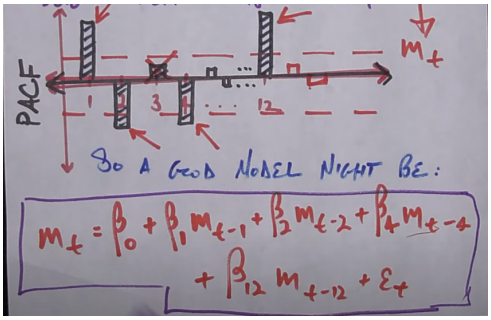
$$(1 - \phi_1 L - \phi_2 L^2 - \phi_3 L^3) y_t = (1 - \theta_1 L - \theta_2 L^2 - \theta_3 L^3)\varepsilon_t$$

$$\Phi(L) y_t = H(L)\varepsilon_t$$

8.8 Autoregressive Model

- Autoregressive that means its a regression, so we are trying to predict something based on past values of the same thing
- In statistics if there are 2 models that have the same behavior we are going to prefer the simplest one (Occam's razor)
- notations for the exercise: m_t quantity of milk per month, m_{t-12} quantity of milk last year
- We want to use only the significative lags in order to predict the quantity of milk, if Im going to use all of them our model will be too tuned and we will get overfitting
- **PACF measure the direct correlation** and not taking into the account the indirect ones ($m_{t-3} \rightarrow m_t$ removing m_{t-2} and m_{t-1}). So we are going to use into the model the band that are statistically different then 0



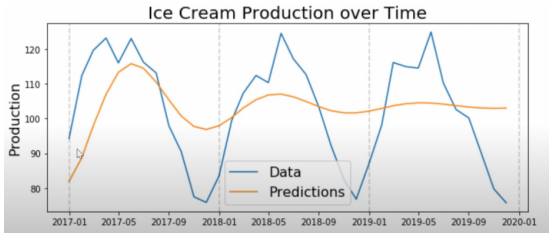


8.9 Evaluate time series model

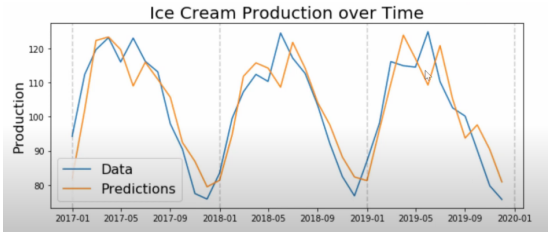
- Run machine learning: **train test split**, we train our model on 70% of the data and test on 30%
- Whenever we have a pattern in the residuals there is some dynamics that we didnt caputure with our model and so there is variability left inside the residuals → not great sign
- We are less and less certain on the predictions we are making moving further into the future and we arrive to predict essentially the mean of the time series. So splitting the data into training and testing we will see that the predictions will be okay initally and drop to mean widening the window of the prediction
- we can calculate the **mean absolute percent error**
- **Rolling forecast origin:** predict one month in advance each time and then average all the predictions. We need to fit the model every single time

Train on months: $1, 2, \ldots, k-3$	***Predict month*** $k-2$
Train on months: $1, 2, \ldots, k-3, k-2$	***Predict month*** $k-1$
Train on months: $1, 2, \ldots, k-3, k-2, k-1$	***Predict month*** k

average all the predictions



splitting dataset into train and test



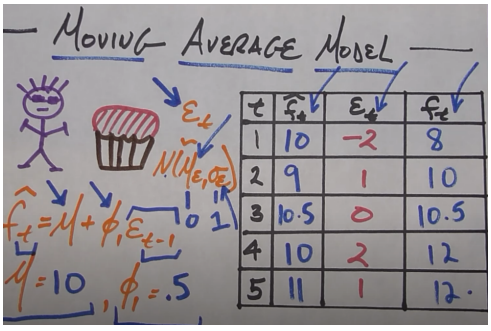
Rolling forecast origin

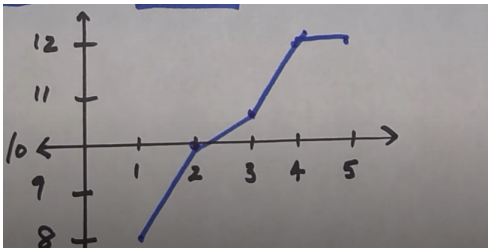
9 Moving average model

- $\hat{f}(t)$ predicted number of cupcakes
- ε_t error the professor tells u
- $f(t)$ how many cupcakes u should have bought that month
- this trend seems to be centered in 10. This is why this model is called moving average, because we have our average of 10 but that average its moving all about that average but still staying centered sort of over there → always center at 10
- The model we are seeing its a MA(1) so we are taking into account one error from the last month
- In order to estimate the model from the data check ACF function and see the correlated lags
- μ is a additive constant, in our IMAD models is 0

$$MA(1) : \hat{f}_t = \mu + \phi_1 \varepsilon_{t-1} \qquad \varepsilon_t \sim N(\mu_\varepsilon, \sigma_\varepsilon^2) = (0, 1)$$
$$\mathbb{E}[\hat{f}_t] = \mathbb{E}[\mu + \phi_1 \varepsilon_{t-1}] = 10 + 0$$

$$MA(2) : f_t = \mu + \phi_1 \varepsilon_{t-1} + \phi_2 \varepsilon_{t-2} + \varepsilon_t$$
$$\hat{f}_t = \mu + \phi_1 \varepsilon_{t-1} + \phi_2 \varepsilon_{t-2}$$

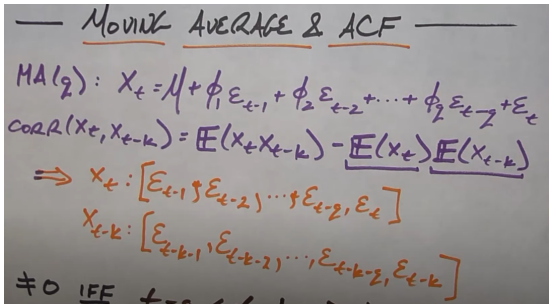


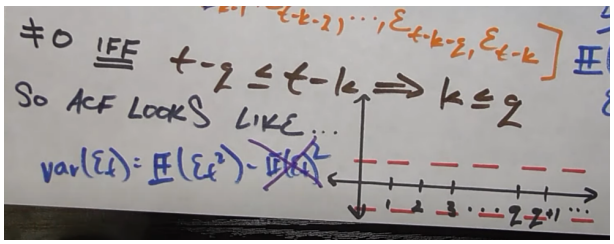


9.1 Moving Average and ACF(Auto correlation function)

- In the correlation fuction the only terms that will be not equal to 0, will be the variance $\mathbb{E}[\varepsilon_t^2]$ this will happen if we have overlapping lags
- $\mathbb{E}[\varepsilon_t \cdot \varepsilon_{t-k}] = 0, k \neq 0$, because the errors are indipendent

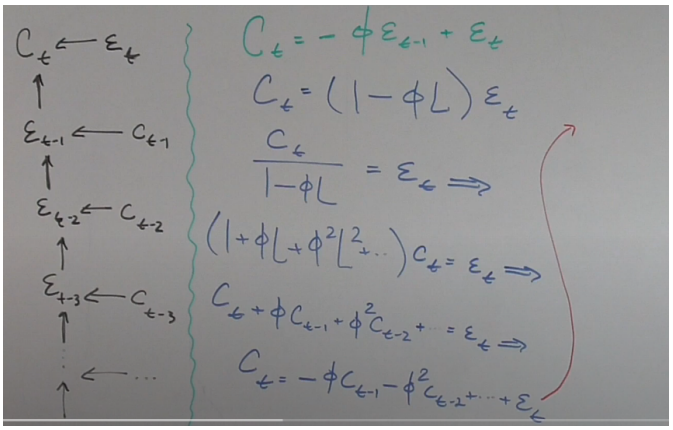
$$ACF(k) = corr(x_t, x_{t-k}) = \mathbb{E}(x_t, x_{t-k}) - \mathbb{E}(x_t)\mathbb{E}(x_{t-k})$$





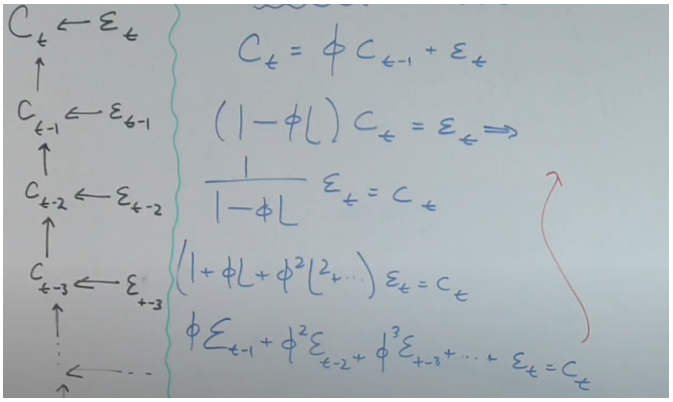
9.2 Invertibility of Time Series: $MA(1) \Leftrightarrow AR(\infty)$

- Connection between AR and MA models
- $\frac{1}{1-\phi L}$ is an infinite geometric sum, where the common ratios is ϕL, the restriction is that the common ratios absolute value must me less than 1 if not the geometric series doesnt converge $\rightarrow$ $|\phi| < 1$ in this way we can say that the time series is invertible



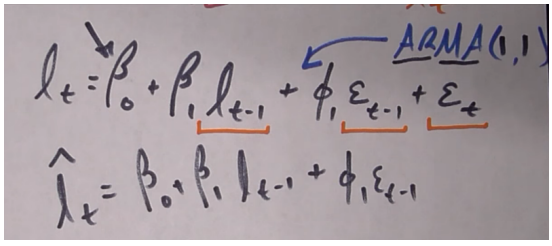
9.3 Invertibility of Time Series: $AR(1) \Leftrightarrow MA(\infty)$

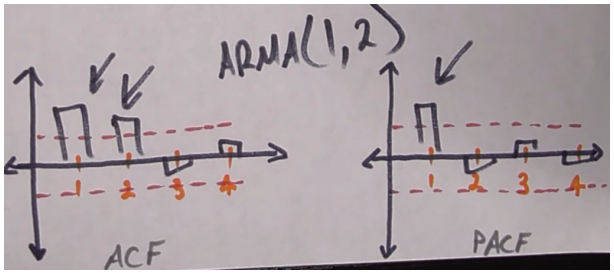
- Same notes as before: important is that $|\phi| < 1$
- so intertibility proccesses are not seprate entities but are the two sides of the same coin



9.4 ARMA model

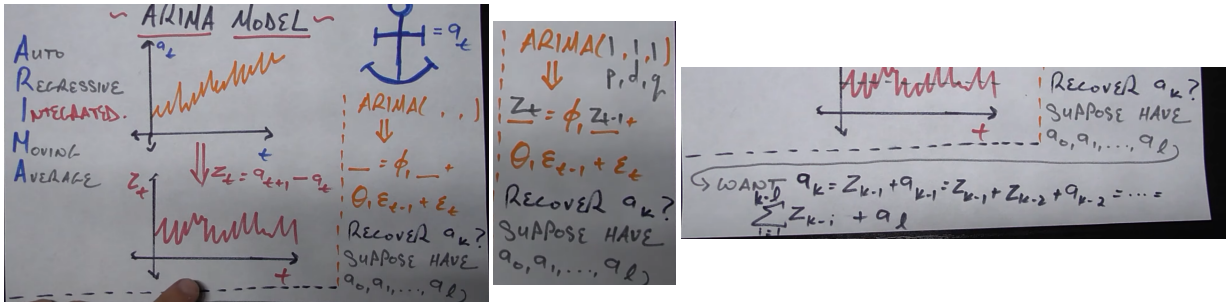
- l_t : light bulbs demand at the current time, so how many should I build up
- β_0 : cosntant demand of light bulbs
- $\beta_1 l_{t-1}$ light bulbs needed to create last month
- $\phi_1 \varepsilon_{t-1}$: error from the previous time period. Last month I made some prediction of how much light bulbs to create and my prediction was off something
- ACF helps to indentify the MA part and PACF the AR part of the ARMA





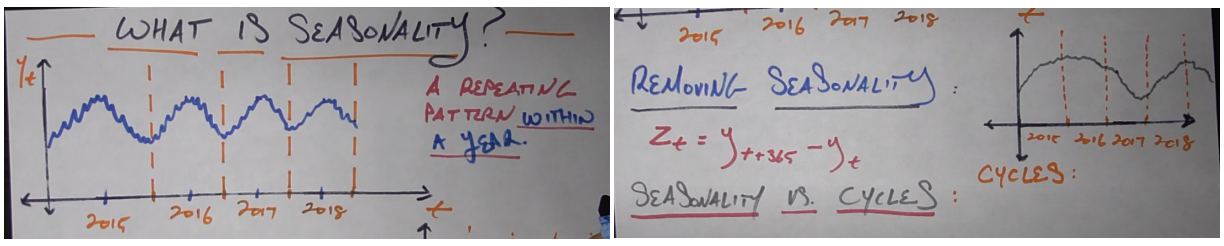
9.5 ARIMA model

- q_t : number of anchor u sell every month
- we can't use ARMA model because the time series is not stationary because of a linear trend in the mean → in this case we can use ARIMA models
- **Integrated:** instead of predicting the time series itself we are going to predict the differences from one timestamp to the previous one. So we are basically creating a new time series which is z_t
- Basically a function is going to a precise pattern and this pattern is constant, so if the fucntion is going to increase +1 at $t = 0$, is going to do the same at $t = 1$ and so one. The behavior of a function is constant over time as well as our time series, and thats why we have created the new time series z_t
- Setting the parameter $d = k$ we are going to differentiate it k times in a row getting a new time series. Im going to stop the differentiate (and so the parameter d) proccess once the time series is stationary. We want d to be as low as we can in order to build up a simple model
- What we want to predict is not the diffrence but the actual time series, so how we get back from z_t to q_t: (q_l last value we have data for)



9.6 What is Seasonality

- **Seasonality** is a repeating pattern within a year that is predictable, instead the **cycles** is a pattern that shows up during the years and is an unpredictable component
- If a time sieres has a seasonal component it isnt stationary so we cant use any model we know so far.
- **Idea:** if im going to take in each period a value in in the same position I'm expecting to get the same value (seasonality - repeating pattern). In order to remove it im going to delete the current value in the period with the previous one and get the innovation out of it



10 Roadmap time series

- Visualizing the time series
- distribution of the covariates
- Uderstand the imputation uncertainty
- response variable NH3 concentration
- use ACF and PACF on a subset pf the data (there is a function inside ACF that can skip the NA values)
- preliminary analisys on the missing data
- Cleaning dataset:
  - replace NAs with 0
  - replace missing values with average values
  - remove Nas values
  - missing robost method
- ARIMA doesnt have any problem with missing values, is a resistent estimation method on missing data without any assumpotion on them
- Model selection techniques: AIC, BIC, lasso, ridge
- use only the significative covariates and standardize it so will me in the same units
- Decompose the model into:

$$y(t) = trend + seasonality + Cyclicity + innovation$$

$$seasonality:\ additive\ or\ multiplicative$$

$$detrending$$

- Make the time series stationary and test the stationary
- Model the innovation using multiple models
- Validation of the dataset 70% and 30% and try other techniques, or block boostrap (there is some link somewhere on the teams)
- Block boostrap only if the data is stationary
- Residuals check $e(t) \sim Wn$
- Compare models and choose the best one according to the Occam's razor
- try machine learning methods and deep learning ones and explain the theory behind ht method
  - Kaggle
  - LightGBM
  - Decision Trees
  - XGBoost
  - Long Short-Term Memory (LSTM)
  - Recurrent Neural Network (RNN)
  - Multi-Layer Perceptron (MLP)

$$site - link$$

- We can even delete the time component and deal with a regression model
- If we use auto arima we have to read the paper and see how it works
- Use RMSE in order to get the best model

11 Machine learning models

- **Ensemble learning:** combine multiple weak models/learners into one predictive model to reduce bias, variance and/or improve accuracy. These models are known as **weak learners**. The intuition is that when you combine several weak learners, they can become **strong learners**.
- **Gradient Boosting** is an ensemble learning algorithm used for both regression and classification tasks. Here's a brief description of how it works:
  - **Base Learners:** Gradient Boosting builds an ensemble of weak learners (usually decision trees) sequentially. Each weak learner tries to correct the errors made by the previous ones.
  - **Gradient Descent:** It optimizes a loss function by adding weak learners to the ensemble in a forward stage-wise manner. At each stage, the algorithm calculates the gradient of the loss function with respect to the ensemble's prediction, and fits a weak learner to the gradient.
  - **Boosting:** The subsequent weak learners focus on the mistakes (residuals) made by the previous ones. They are trained on the residuals of the previous learners rather than the original data.
  - **Weighted Voting:** The final prediction is made by a weighted sum of the predictions of all weak learners. The weights assigned to each learner depend on their individual performance in minimizing the loss function.
  - **Regularization:** Gradient Boosting employs regularization techniques to prevent overfitting, such as limiting the maximum depth of the trees, adding a shrinkage parameter to the predictions, and using subsampling of the data.
- **XGBoost:** An optimized distributed library for machine learning models in the gradient boosting framework, designed to be highly efficient, flexible, and portable. It features regularization parameters to penalize complex models, effective handling of sparse data for better performance, parallel computation, and more efficient memory usage

11.1 XGBoost

- XGBoost (eXtreme Gradient Boosting) is an implementation of the Gradient Boosting algorithm
- XGBoost is a decision tree-based ensemble algorithm that leverages the concept of gradient boosting to create a strong model
- **How it works:** works by creating a set of decision trees iteratively, each tree attempting to correct the mistakes of the previous tree. The algorithm employs a **gradient descent algorithm** to minimize a cost function, which is the sum of the errors of each tree in the ensemble.
- The final model is a weighted combination of all the decision trees, with each tree assigned a weight based on its contribution to the cost function.
- Steps:
  1. **Initialize the model:** the first step in XGBoost is to initialize the model by creating a base prediction. This prediction is usually the mean of the target variable for regression problems or the most common class for classification problems.
  2. **Fit the first tree:** to train the first tree of a model, features and residuals are used in a greedy manner where informative features are selected first. The residuals are the differences between the model's predictions and the actual values.
  3. **Compute the loss:** after training the first tree, the loss is computed by summing the errors made by all trees in the ensemble. The sample weights are then updated using the loss, with higher weights assigned to samples with higher errors.
  4. **Fit the next tree:** after computing the loss, the next step is to fit the next tree. This tree is trained on the updated weights and the residuals from the previous tree. The process is repeated until a predetermined number of trees is reached or the loss stops decreasing.
  5. **Make predictions:** The final step in XGBoost is to make predictions using the ensemble of trees. The prediction for each sample is the weighted sum of the predictions made by each tree in the ensemble.
- XGBoost utilizes several techniques to build its algorithm. These include:

- **Tree Building:** XGBoost uses a greedy algorithm to build decision trees. It starts with a single node and recursively splits the data into two child nodes based on the feature that maximizes the reduction in the loss function. The loss function is typically the negative log-likelihood for classification problems and the mean squared error for regression problems. The splitting process continues until a stopping criterion is met, such as a maximum depth or a minimum number of samples per leaf node. XGBoost also includes **tree pruning** to prevent overfitting, which removes nodes that do not contribute to the reduction in the loss function.
- **Gradient Descent:** gradient descent is an iterative method to find the minimum of a function by adjusting the parameters towards the negative gradient. In XGBoost, the loss function is the sum of the losses of each data point and a regularization term that penalizes large coefficients. The regularization term, either L1 or L2, makes the model simpler and prevents overfitting. interpretable.

11.2 Code:

In the context of XGBoost, the terms "Gain," "Cover," and "Frequency" represent different aspects of feature importance calculated by the algorithm:

Gain: This refers to the improvement in accuracy brought by a feature to the branches it appears in. Essentially, it measures the contribution of each feature to the model's performance. Higher gain values indicate that the feature is more important for making decisions in the model.

Cover: Cover stands for the relative quantity of observations concerned with a specific feature. It's the sum of second-order gradients for all instances in a tree that the feature is used to split. Higher cover values indicate that the feature is used to make more splits across the dataset.

Frequency: Frequency denotes the number of times a feature appears in all generated trees. It provides insights into how often a feature is used for splitting nodes across all the trees in the ensemble.

In summary, while Gain assesses the quality of a feature, Cover and Frequency offer information about the quantity and usage of the feature in the construction of the ensemble model. These metrics collectively help in understanding the importance and contribution of each feature towards the predictive performance of the model.

11.3 Understanding Facebook's Prophet

Prophet is a forecasting tool developed by Facebook's Core Data Science team. It's designed to make time series forecasting tasks more accessible to analysts and non-experts. Here's how it works:

1. **Automatic Forecasting**: Prophet is built to handle time series data with strong seasonal patterns and multiple seasonality. It automatically detects these patterns in the data and fits a model accordingly.

2. **Flexible Model**: Prophet uses a decomposable time series model with three main components: trend, seasonality, and holidays. The trend component models non-periodic changes over time, while the seasonality component captures periodic changes (daily, weekly, yearly). Holidays and special events can also be incorporated into the model.

3. **Bayesian Approach**: Prophet uses a Bayesian approach to model fitting, which allows for uncertainty estimation in the forecasts. This means that Prophet provides not only point forecasts but also uncertainty intervals around the predictions, which can be valuable for decision-making.

4. **Robustness to Missing Data and Outliers**: Prophet is designed to handle missing data and outliers in the time series gracefully. It employs a procedure called "imputation" to fill in missing values in the data, and it uses a robust method to identify outliers and mitigate their impact on the forecasts.

5. **User-Friendly Interface**: One of the key features of Prophet is its user-friendly interface. It provides a simple API that allows users to specify the time series data, as well as any additional information such as holidays or seasonality patterns. This makes it easy for analysts and data scientists to use Prophet even without extensive knowledge of time series forecasting techniques.

6. **Scalability**: Prophet is designed to be scalable and can handle large datasets efficiently. It's implemented in Python and is built on top of the Stan probabilistic programming language, which provides efficient algorithms for Bayesian inference.

Max deapth	**min iterations**	**train** $RMSE$	**validation** $RMSE$
`max_deapth_1`	998	6.088	7.325
`max_deapth_2`	997	5.349	7.132
`max_deapth_3`	745	4.552	7.097
`max_deapth_4`	324	3.684	7.047
`max_deapth_5`	336	2.758	6.943
`max_deapth_6`	347	1.914	6.954

Model	**weekly**	**monthly**	**annual**
`XGBoost`	2.200e-16	2.200e-16	2.200e-16
`Prophet`	2.200e-16	2.200e-16	2.200e-16
`LSTM`	0.058	0.355	0.065

$$y(t) = g(t) + s(t) + h(t) + e(t)$$

Model	**train** $RMSE$	**validation** $RMSE$	**Shapiro test**	**Breusch-Pagan Test**
`XGBoost`	4.281	6.934	1.227e-08	0.186
`Prophet`	6.772	7.924	4.721e-09	0.240
`LSTM`	1.238	7.195	0.062	0.207e-03

Epochs	**train** $RMSE$	**validation** $RMSE$	**Shapiro test**	**Breusch-Pagan Test**
200	1.331	7.010	2.772e-05	0.253e-03
100	1.238	7.195	0.062	0.207e-03
50	2.223	7.173	4.409e-05	1.556e-05

Model	**weekly**	**monthly**	**annual**
200	0.025	0.543	0.299
100	0.058	0.355	0.065
50	2.116e-05	0.018	0.373