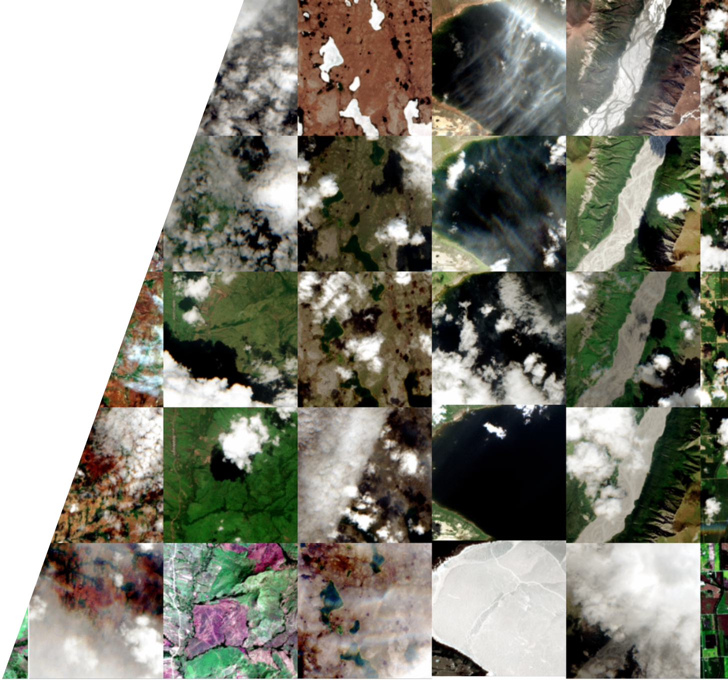
How reliable are Sentinel-2 cloud detection algorithms?

Global uncertainty estimation with gaussian processes.

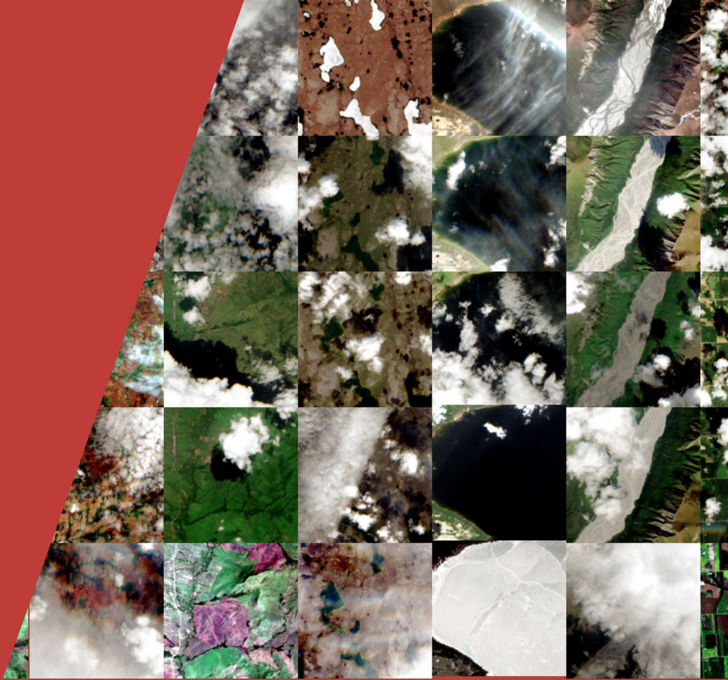
Cesar Aybar

April 11, 2022

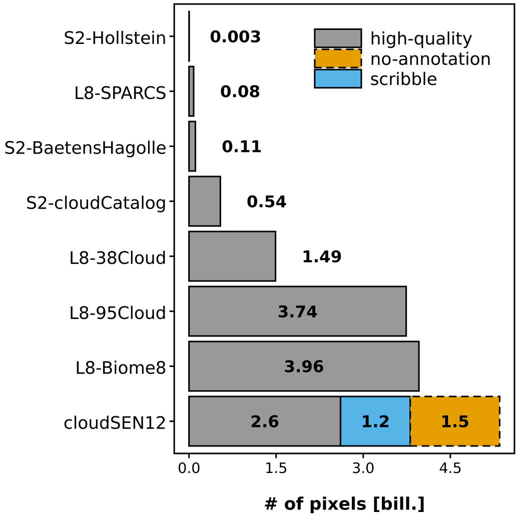


Context

- Available cloud datasets
- Shortcomings

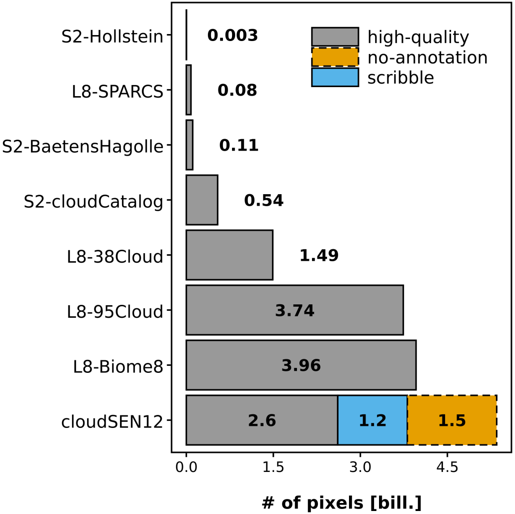


Available cloud datasets



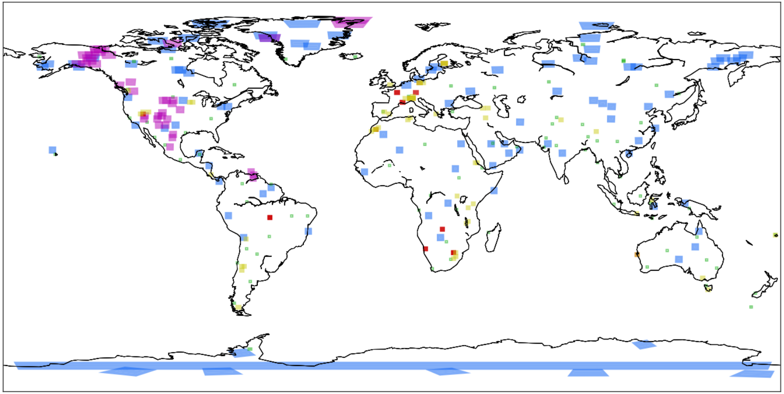
- Cloud labels created by human photo-interpretation.

Available cloud datasets



- Cloud labels created by human photo-interpretation.
- There are just three cloud shadow providers: Hollstein, SPARCS, and **cloudSEN12**.

Available cloud datasets



L8-Biome [33]   L8-SPARCS [34]   L8-38Clouds [20]

S2-Hollstein [32]   S2-BaetensHagolle [36]

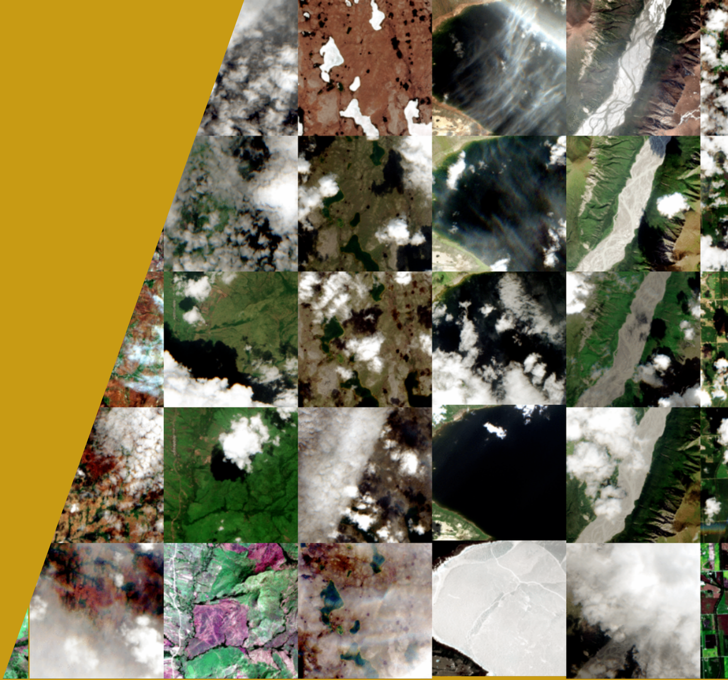
Shortcomings

- None of them have a **time dimension**.
- **Downloading is a struggle**. Using STAC is a no-brainer :).
- As is often with EO datasets, no information regarding the **control quality** is provided.
- **Human level performance** is not evaluated.
- High class imbalance.
- Geographically biased.
- The creation procedure is unclear.

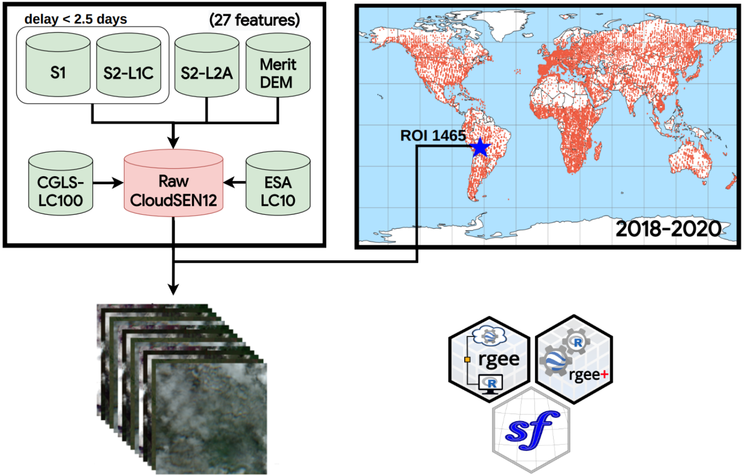
Shortcomings

cloudSEN12

- Data Preparation
- Image Patch Selection
- Labeling



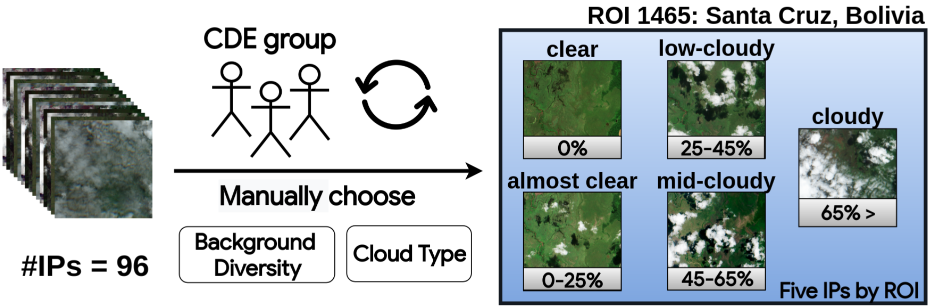
Data Preparation



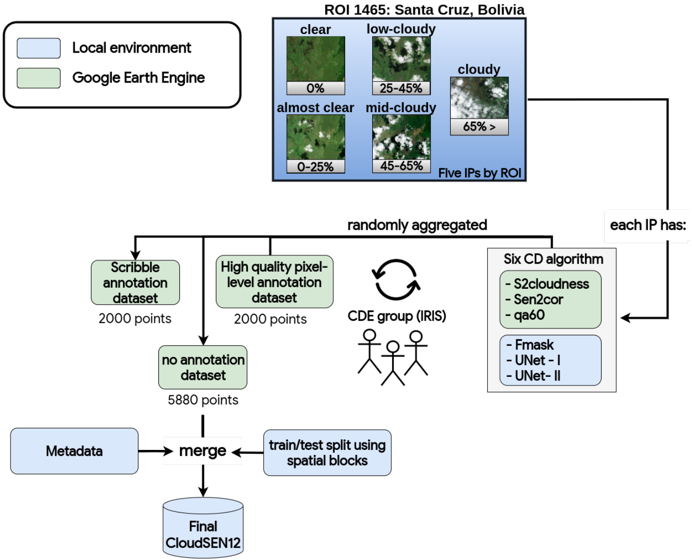
Ready for **tile selection!**

<T, B, H, W>

Image Patch Selection

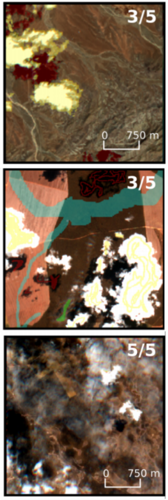


Labeling



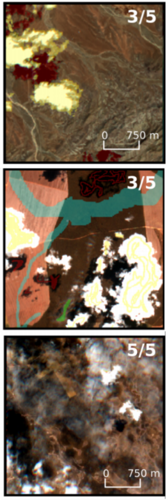
Local environment
Google Earth Engine
ROI 1465: Santa Cruz, Bolivia
clear
0%
low-cloudy
25-45%
cloudy
65% >
almost clear
0-25%
mid-cloudy
45-65%
Five IPs by ROI
each IP has:
randomly aggregated
Scribble annotation dataset
2000 points
High quality pixel-level annotation dataset
2000 points
CDE group (IRIS)
Six CD algorithm
- S2cloudness
- Sen2cor
- qa60
- Fmask
- UNet - I
- UNet- II
no annotation dataset
5880 points
Metadata
merge
train/test split using spatial blocks
Final CloudSEN12

Labeling



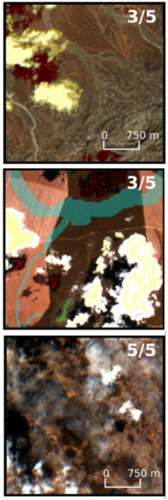
- **2000 ROIs** with pixel level annotation, where the average annotation time is 150 minutes (high-quality group).

Labeling



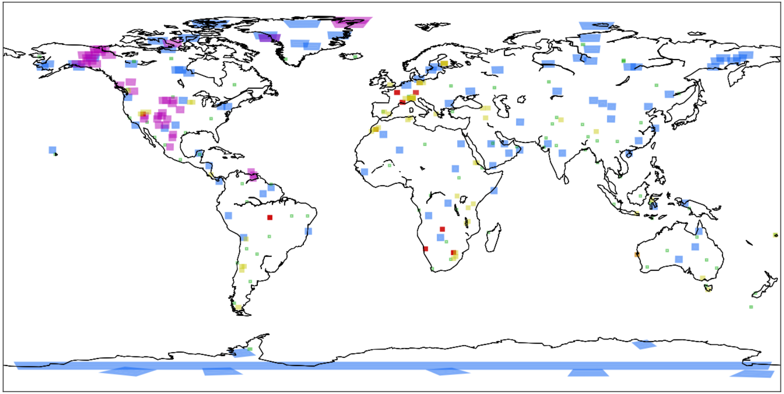
- **2000 ROIs** with pixel level annotation, where the average annotation time is 150 minutes (high-quality group).
- **2000 ROIs** with scribble level annotation.

Labeling



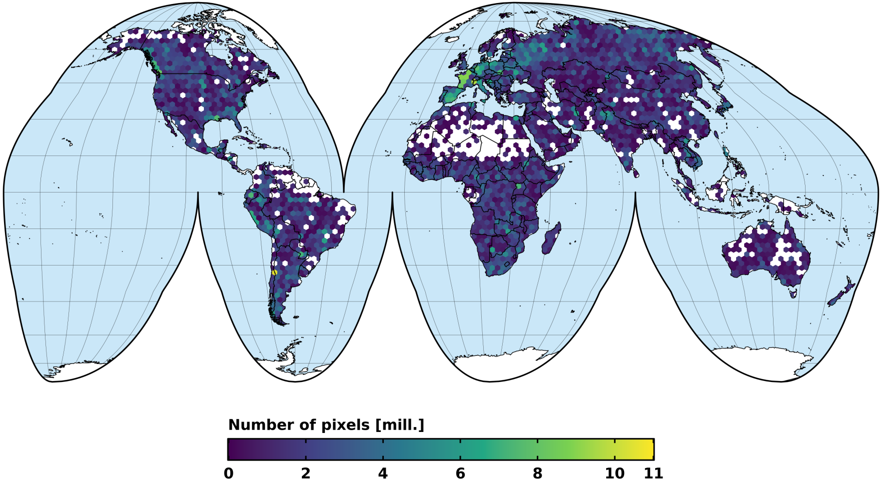
- **2000 ROIs** with pixel level annotation, where the average annotation time is 150 minutes (high-quality group).
- **2000 ROIs** with scribble level annotation.
- **5880 ROIs** with annotation only in the cloud-free (0%) image (no annotation group).

Data Preparation



L8-Biome [33] L8-SPARCS [34] L8-38Clouds [20]
S2-Hollstein [32] S2-BaetensHagolle [36]

Data Preparation

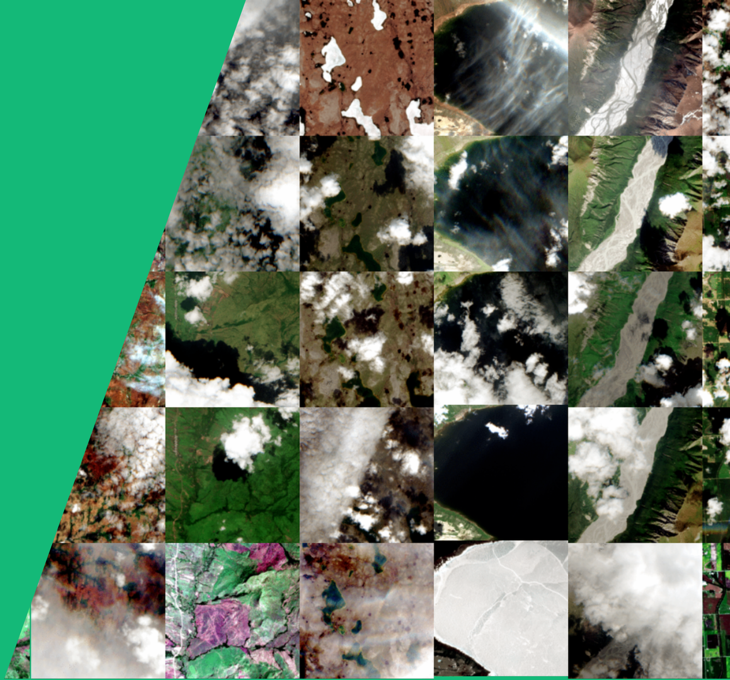


The website

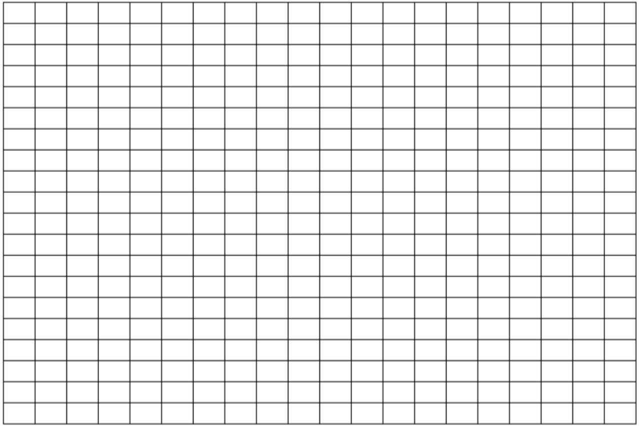
https://cloudsen12.github.io/

Uncertainty estimation

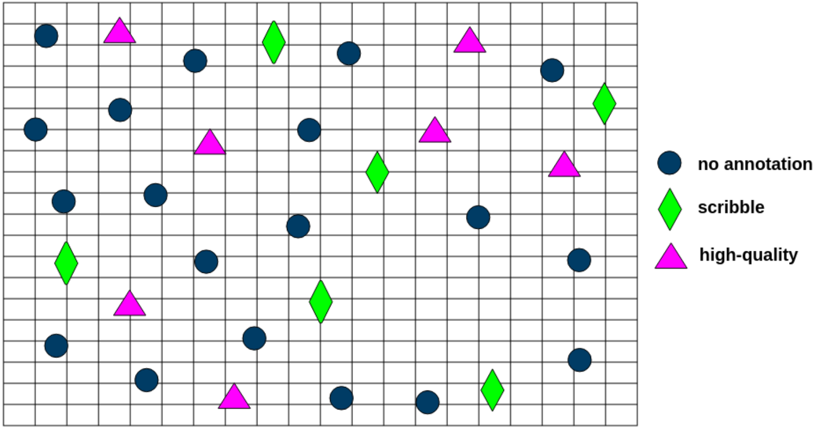
- The problem.
- Gaussian proccess
- Locally Weighted Gaussian Process
- 3D-kernel
- TODO



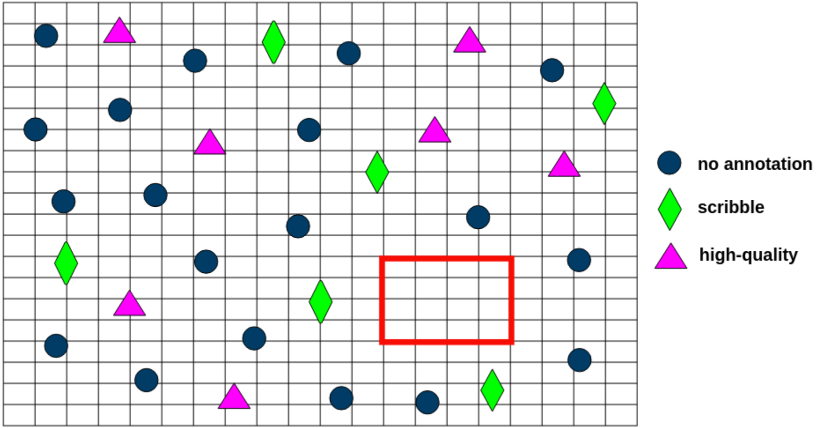
The problem



The problem



The problem



The problem

- How do the models perform spatially?
- Is there a correlation between model performance and cloud cover?
- How do the models do in the summer compared to the winter?
- How do the models perform at cloud borders?
- How do techniques based on physics and machine learning approaches compare?

Can we build a model that answers all of these?

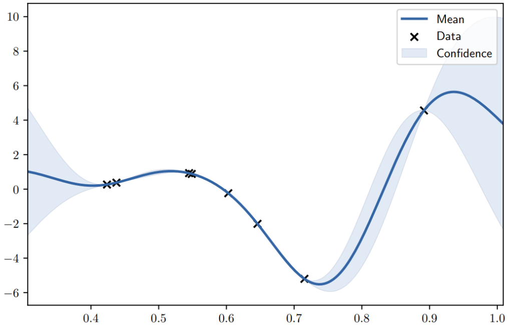
Gaussian process

"Gaussian process can do everything that NN do but better"

Input and Output Data:

$$\mathbf{y} = (y_1, \ldots, y_N), \quad \mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^\top$$

$$p(\mathbf{y} \mid \mathbf{f}) = \mathcal{N}\left(\mathbf{y} \mid \mathbf{f}, \sigma^2 \mathbf{I}\right), \quad p(\mathbf{f} \mid \mathbf{X}) = \mathcal{N}(\mathbf{f} \mid 0, \mathbf{K}(\mathbf{X}, \mathbf{X}))$$



Gaussian process

- Maximum likelihood estimate of the hyper-parameters.

$$\theta^* = \arg\max_{\theta} \log p(\mathbf{y} \mid \mathbf{X}, \theta) = \arg\max_{\theta} \log \mathcal{N}\left(\mathbf{y} \mid 0, \mathbf{K} + \sigma^2 \mathbf{I}\right)$$

- Prediction on a test point given the observed data and the optimized hyper-parameters.

$$p\left(\mathbf{f}_* \mid \mathbf{X}_*, \mathbf{y}, \mathbf{X}, \theta\right) = \mathcal{N}\left(\mathbf{f}_* \mid \mathbf{K}_*\left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y}, \mathbf{K}_{**} - \mathbf{K}_*\left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{K}_*^{\top}\right)$$

Gaussian process

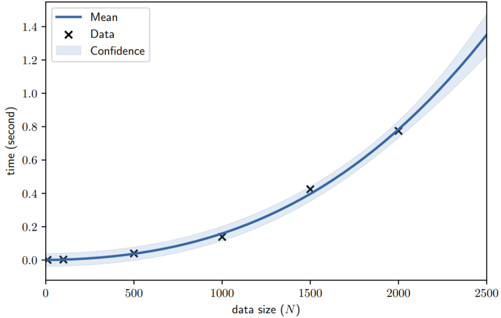
- Plug in the log-pdf of multi-variate normal distribution:

$$
\begin{aligned}
\log p(\mathbf{y} \mid \mathbf{X}) &= \log \mathcal{N}\left(\mathbf{y} \mid 0, \mathbf{K} + \sigma^2 \mathbf{I}\right) \\
&= -\frac{1}{2} \log \left|2\pi \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)\right| - \frac{1}{2} \mathbf{y}^\top \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y} \\
&= -\frac{1}{2} \left(\left\|\mathbf{L}^{-1}\mathbf{y}\right\|^2 + N \log 2\pi\right) - \sum_i \log \mathbf{L}_{ii}
\end{aligned}
$$

- Take a Cholesky decomposition: $\mathbf{L} = \mathrm{chol}\left(\mathbf{K} + \sigma^2 \mathbf{I}\right)$.
- The computational complexity is $O\left(N^3 + N^2 + N\right)$. Therefore, the overall complexity including the computation of $\mathbf{K}$ is $O\left(N^3\right)$.

Gaussian process

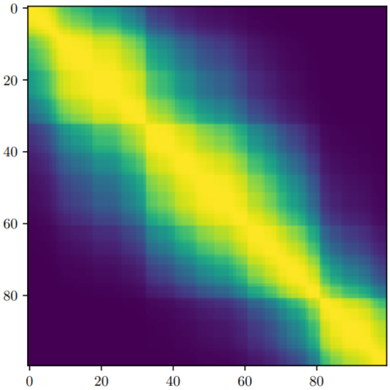
- I collect the run time for $N = 10, 100, 500, 1000, 1500, 2000$.
- They take $1.3ms, 8.5ms, 28ms, 0.12s, 0.29s, 0.76s$.



- For 18000 points estimate the hyper-parameters will take approximately 27 hours.

Gaussian process

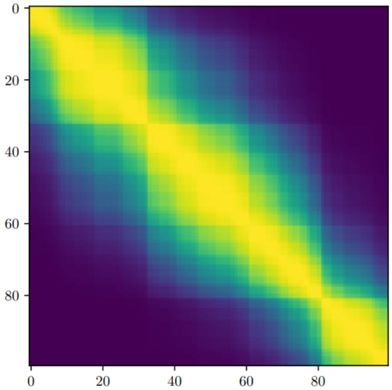
With redundant data, the covariance matrix becomes low rank.



- Low rank Gaussian Process.

Gaussian process

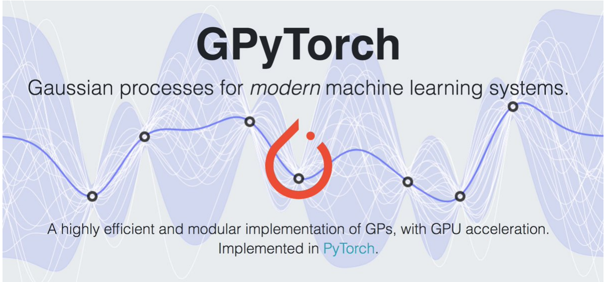
With redundant data, the covariance matrix becomes low rank.



- Low rank Gaussian Process.
- Sparse Gaussian Process.

Scalable Gaussian process

- BlackBox Matrix-Matrix Inference (BBMM).
- LancZos Variance Estimates (LOVE).



https://gpytorch.ai/

Geographically Weighted Gaussian Process

- **Inputs** : Points distributed around the world (X, Y, M).
- **Output** : Predicted values of regression points.
- **Variables:**
  - **bw:** Geographically Bandwidth.
  - **Kn:** Kernel function.
  - **Ln:** Linear Regression parameters.
  - **Cn:** Covariance matrix.

Nonstationary Gaussian Process

Locally Weighted Gaussian Process Approach

```
for each updated sets of reference points Rfp do:
- Create X and Y matrices.
for i = 1 to number of regression points Rn do:
- Calculate W neighborhood matrix using bw, Kn and the distance betwee
- Calculate the regression coefficients (): (i) = (XTW(i)X)1XTW(i)Y
- Estimate the predicted value of yhat for point i using
(i) and Xi (a row vector).
- Obtain the residuals comparing y vs yhat.
- Use a GP to spatially the residuals.
- Sum yhat with the residuals to get the final values.
```

MixedKernel:

The coordinates in our model is set as:

$$(x, y, M)$$

Using Hamming distances we created a combined kernel:

$$K((x1, c1), (x2, c2)) = Kcont1(x1, x2) + Kcat1(c1, c2) + Kcont2(x1, x2) * Kcat2(c1, c2)$$

where xi and ci are the continuous and categorical features of the input, respectively. The suffix i indicates that we fit different lengthscales for the kernels in the sum and product terms.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.
  - **How do the models do in the summer compared to the winter?**: Train a GP splitting first the data by season.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.
  - **How do the models do in the summer compared to the winter?**: Train a GP splitting first the data by season.
  - **How do the models perform at cloud borders?:** Train a GP using just information at borders.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.
  - **How do the models do in the summer compared to the winter?**: Train a GP splitting first the data by season.
  - **How do the models perform at cloud borders?:** Train a GP using just information at borders.
  - **How do techniques based on physics and machine learning approaches compare?:** Compare results between models.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.
  - **How do the models do in the summer compared to the winter?**: Train a GP splitting first the data by season.
  - **How do the models perform at cloud borders?:** Train a GP using just information at borders.
  - **How do techniques based on physics and machine learning approaches compare?:** Compare results between models.
- Using GP and a DL models pretrained in cloudSEN12 we can predict the areas that need more annotations!.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.
  - **How do the models do in the summer compared to the winter?**: Train a GP splitting first the data by season.
  - **How do the models perform at cloud borders?:** Train a GP using just information at borders.
  - **How do techniques based on physics and machine learning approaches compare?:** Compare results between models.
- Using GP and a DL models pretrained in cloudSEN12 we can predict the areas that need more annotations!.
- Considering the variance information we can propose a joined algorithm:

$$dilation(X1 * sen2cor + X2 * s2cloudless + X3 * QA)$$

.

TODO:

- Can the Gaussian Process predict the weights of the local linear model?
- Try to use a non-linear model (NN) to predict the weights.
- The idea is solve the following questions:
  - **How do the models perform spatially?:** Train a GP using the entire dataset.
  - **How do the models do in the summer compared to the winter?**: Train a GP splitting first the data by season.
  - **How do the models perform at cloud borders?:** Train a GP using just information at borders.
  - **How do techniques based on physics and machine learning approaches compare?:** Compare results between models.
- Using GP and a DL models pretrained in cloudSEN12 we can predict the areas that need more annotations!.
- Considering the variance information we can propose a joined algorithm:

$$dilation(X1 * sen2cor + X2 * s2cloudless + X3 * QA)$$

.

- **This would be the first cloud detection algorithm geographically aware!**

Next steps:

- Finish to write the chapter one (cloudSEN12 description).
- Create predictors: X
- Create a tabular dataset with the columns: $x, y, M, cc(target)$.
- Implement the algorithm described above.
- Create a GP model for each question.
- Write chapter two.