

Gaussian Process Regression: An Application in Radio Cosmology

arXiv:2105.12665

Paula Soares

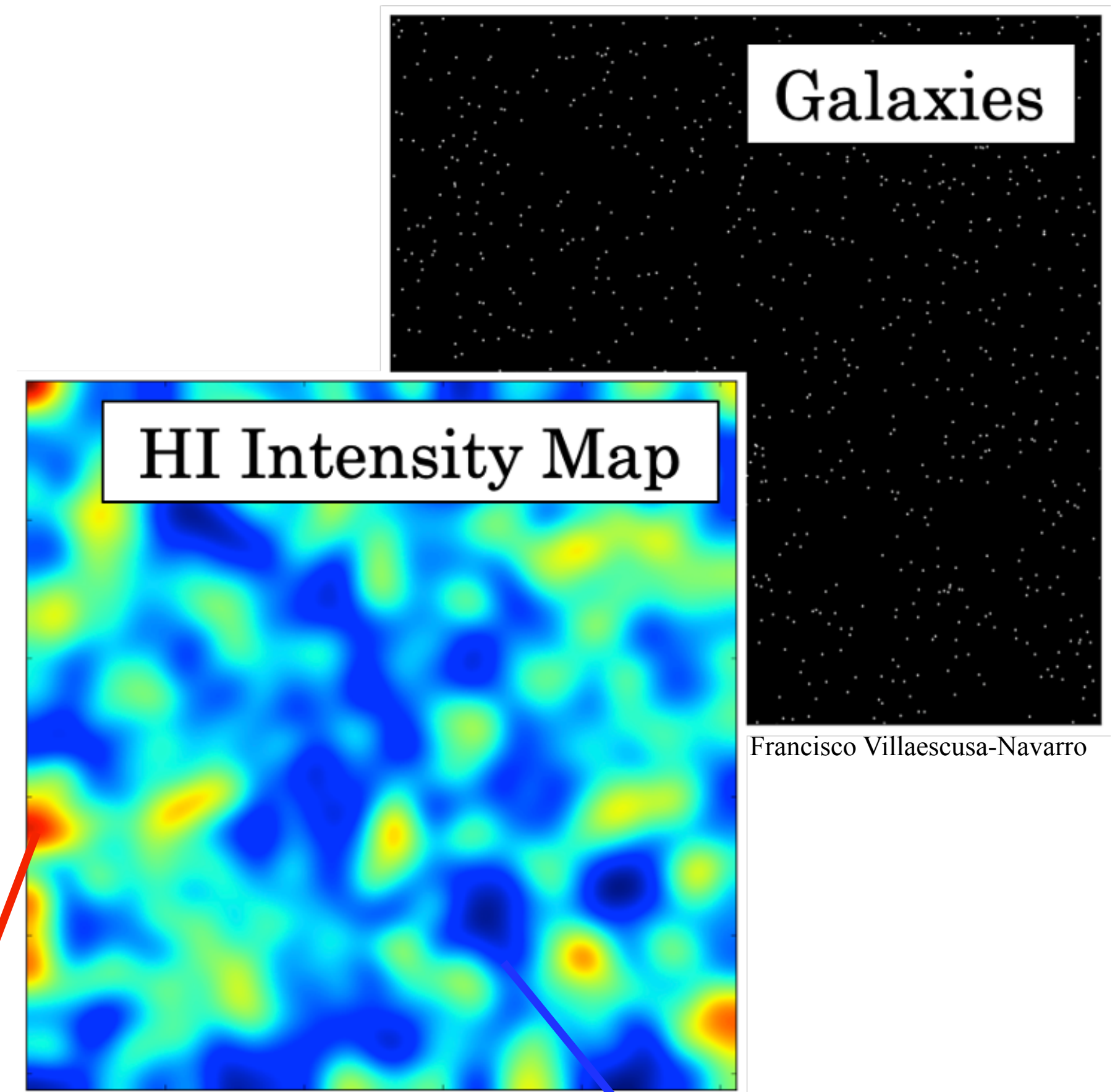
In collaboration with Catherine Watkinson, Steve Cunnington and Alkistis Pourtsidou

Large-scale structure

- How matter is structured on a large scale
- Tells us a lot about our universe, e.g.:
 - Λ CDM parameter and alternatives
 - Physics of dark components
 - Tests to general relativity
- **Galaxy surveys** can trace it, but can be expensive and time consuming

HI intensity mapping

- After reionisation, most of the neutral hydrogen (HI) can be found in galaxies
 - HI is a good tracer of the large-scale structure
- Can quickly map **large** areas of the sky
- Low angular resolution, high frequency resolution

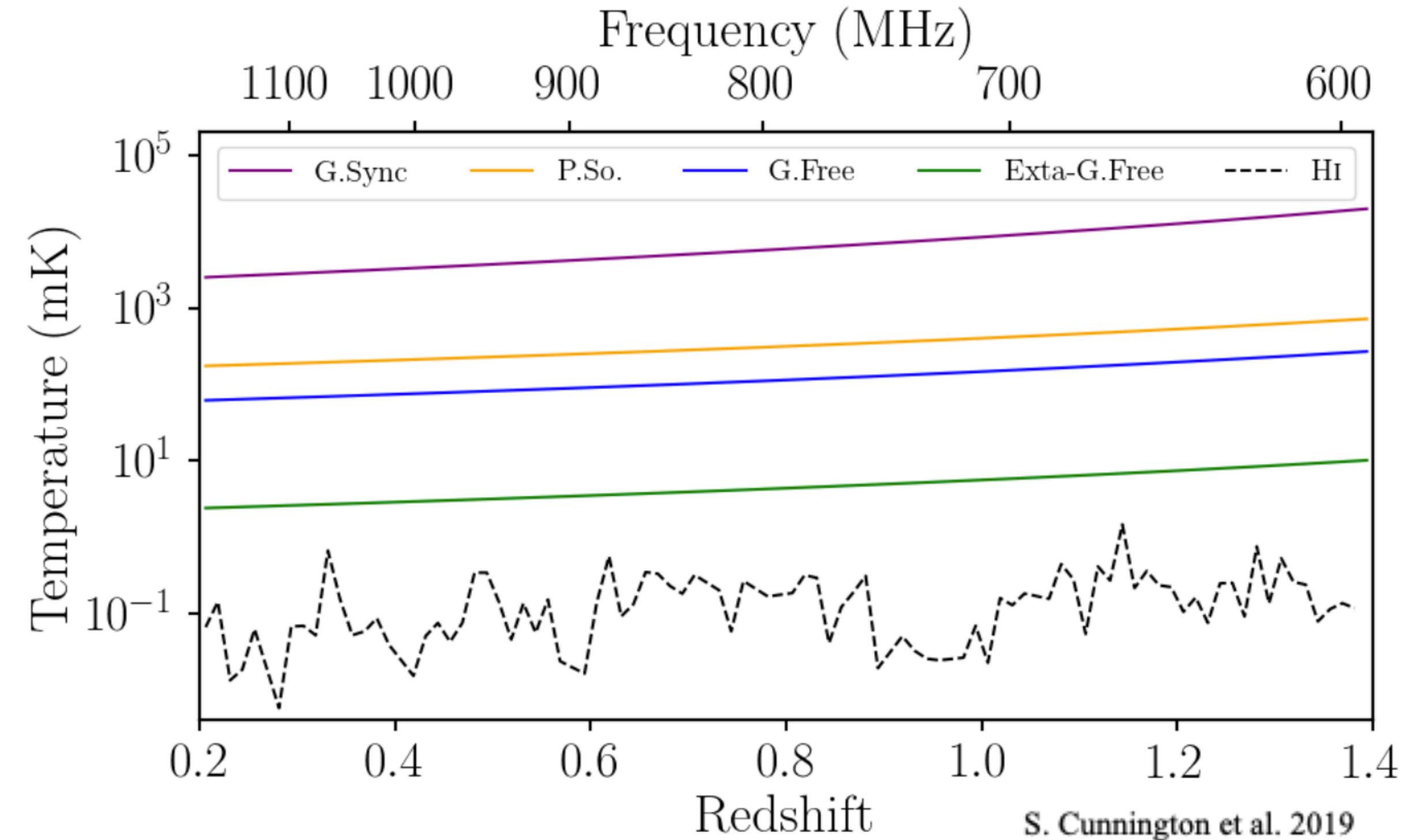


Higher intensity
= more HI present
= more matter present

Lower intensity
= less HI present
= less matter present

The Foreground Problem

- **Foreground:** any other signal we detect which is not the desired HI signal
- Astrophysical foregrounds dominate over the 21cm cosmological signal
 - Galactic synchrotron
 - Point sources
 - Galactic and extra-galactic free-free emission
- They dominate over the HI signal, so **need to be removed**



Foregrounds: bright and smooth in frequency

HI signal: faint and not smooth in frequency

Motivation

- **GPR** has already been applied as a foreground removal technique successfully in the context of the Epoch of Reionisation (see e.g. [Mertens et al. 2018](#) [arXiv:1711.10834] and public code [ps_eor¹](#))
 - * How does **GPR** perform in the case of low redshift, single-dish Intensity Mapping?
 - * How does it compare to other methods e.g. PCA?
 - * Could we use it for future surveys such as the SKA?

¹gitlab.com/flomertens/ps_eor

Gaussian Process

Gaussian Process

A Gaussian process is a Gaussian distribution over infinite dimensions

A Gaussian process is a Gaussian distribution defined by:

- Mean function: $m(\nu) \equiv m$
- Covariance function: $k(\nu, \nu) \equiv K$

$\nu = \text{frequency}$

$$f \sim \mathcal{N}(m, K)$$

\hookrightarrow random variable

Covariance Function

a.k.a. kernel, kernel function, covariance

Typically your covariance function $k(\nu, \nu)$ is itself a function of 3 hyperparameters:

- **Lengthscale** (ℓ): describes how correlated the data is
 - **Variance** (σ^2): describes the amplitude of the signal
 - **Spectral parameter** (η): describes how “smooth” the data is
- Find best fitting values
- Choose a value

Gaussian Process *Regression*

Gaussian Process *Regression*

What is it?

Usually zero!

Assume you have some data (d) which can be describes as a *Gaussian process* with **mean function** $m(\nu)$ and **covariance function** $k(\nu, \nu)$. We can use this to make predictions for what the data would look like at a new frequency (ν'):

Diagram illustrating the Gaussian process model for a function f . The model is represented as a Gaussian distribution over the function values at input points x and x' .

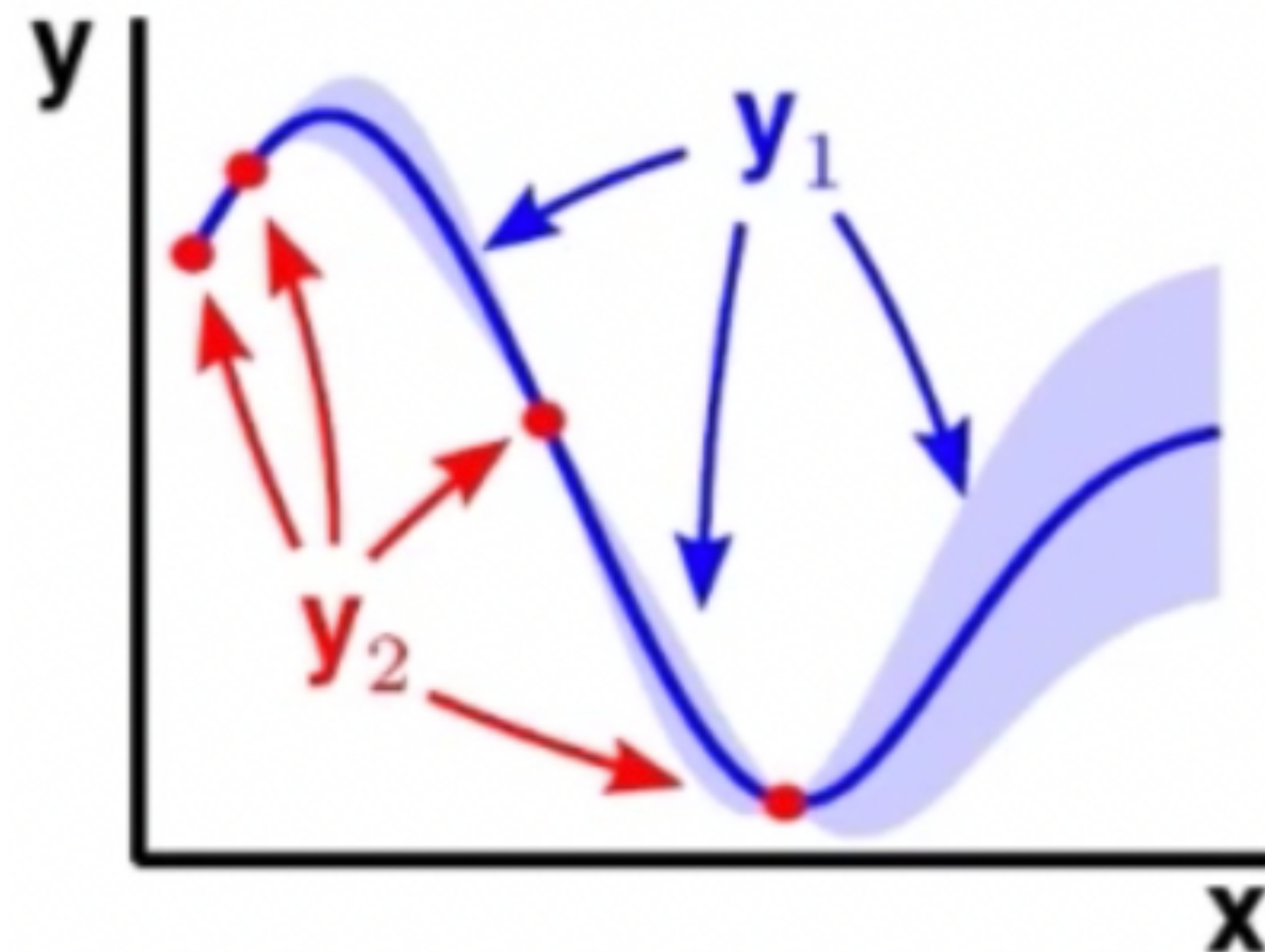
The joint distribution is shown as:

$$\begin{bmatrix} d \\ d' \end{bmatrix} = \mathcal{N} \left(\underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\text{Mean function (zero)}}, \underbrace{\begin{bmatrix} k(\nu, \nu) & k(\nu, \nu') \\ k(\nu', \nu) & k(\nu', \nu') \end{bmatrix}}_{\text{Covariance function}} \right)$$

Annotations:

- d and d' are labeled as "Data points we've already observed".
- The mean function is zero.
- The covariance function is k .

New data points
we want to predict



SKA-like
simulations

**Assume our data, and each of its
components (foreground, H1, noise) is a
Gaussian process**

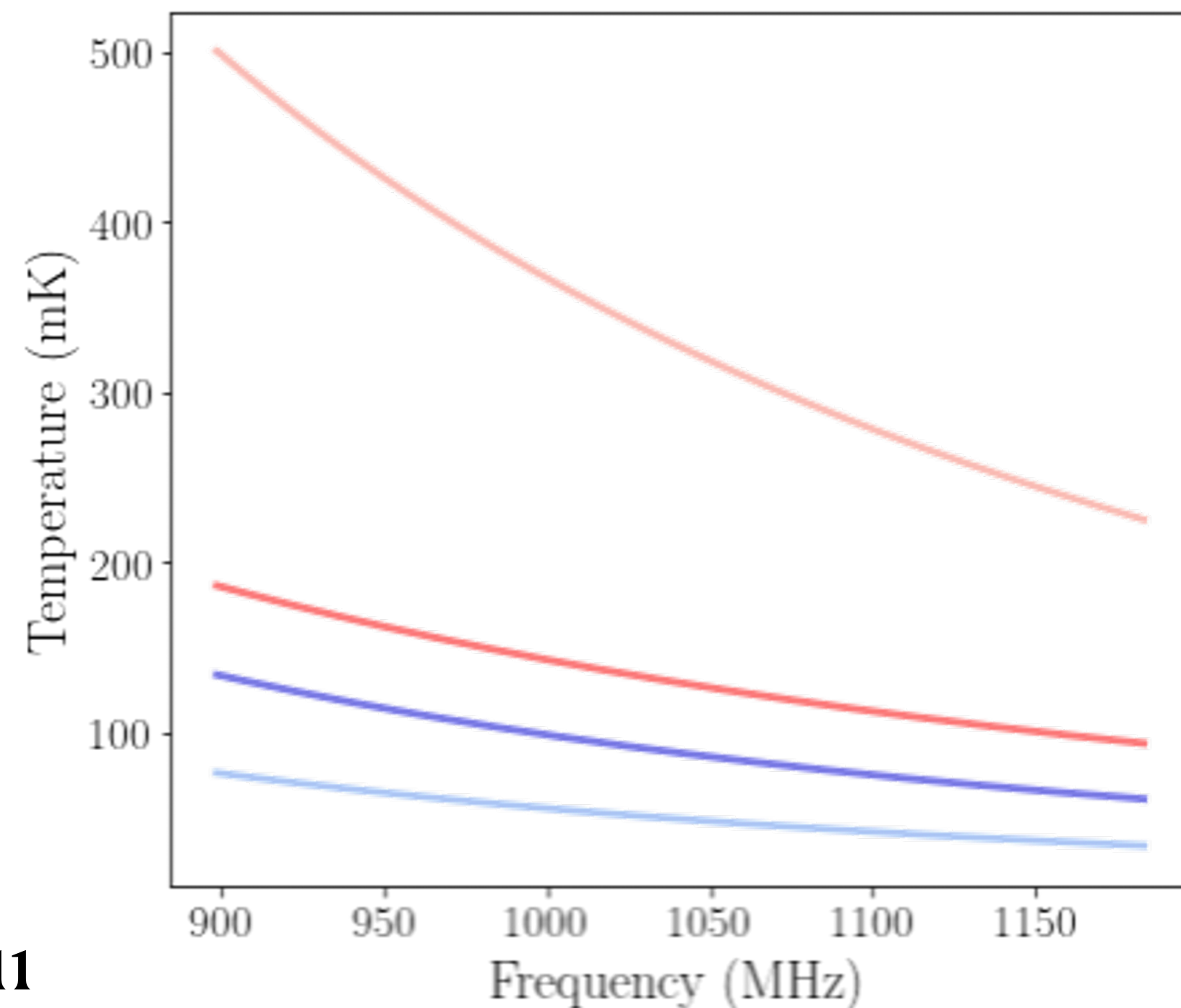
Our data's covariance function:

$$K = K_{\text{fg}} + K_{21} + K_{\text{noise}}$$

$\hookrightarrow K_{\text{fg}} = K_{\text{smooth}} + K_{\text{pol}}$

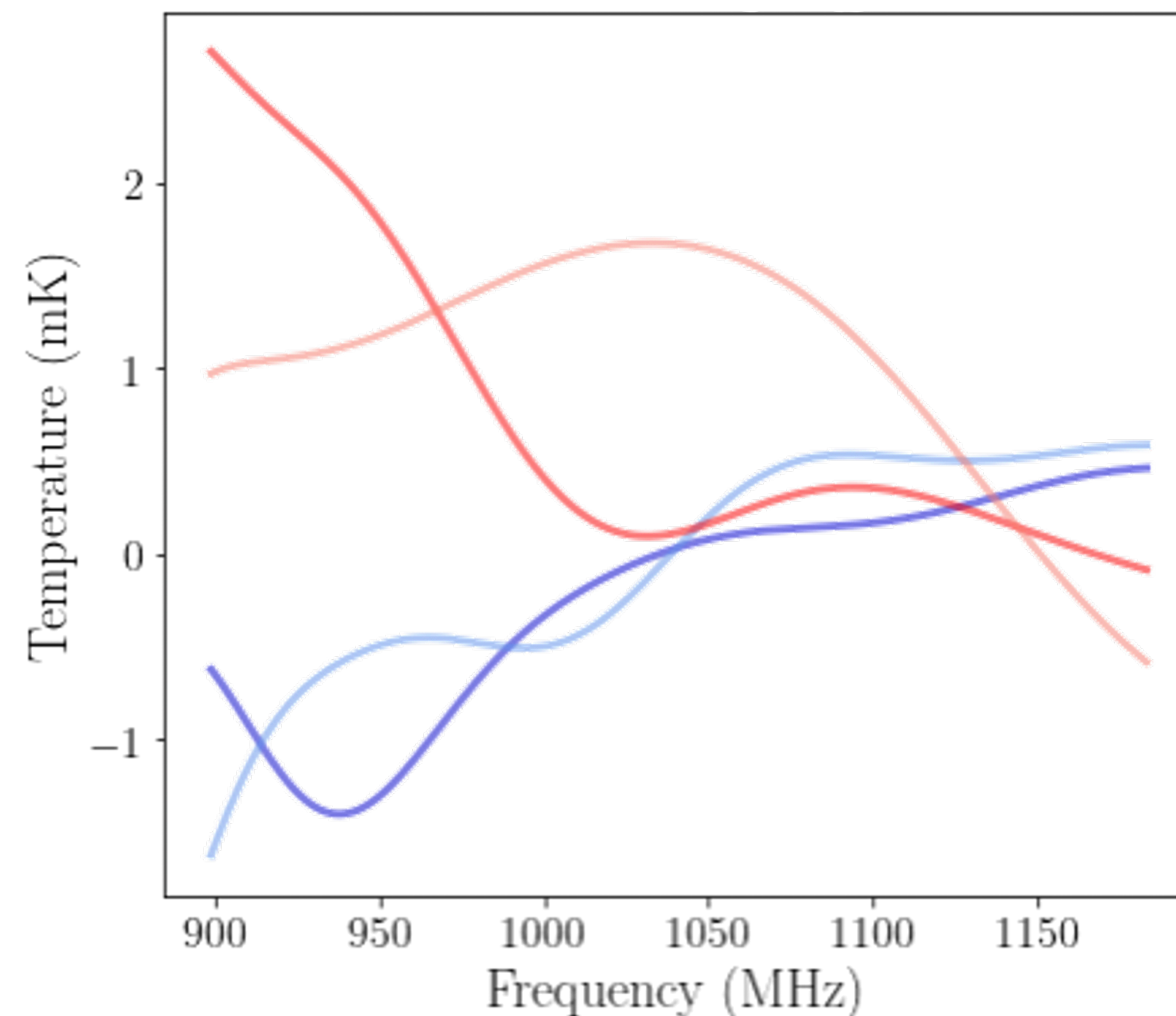
Smooth foregrounds K_{smooth}

- Correlated (large ℓ)
- High amplitude (large σ^2)
- Overall smooth (large η)



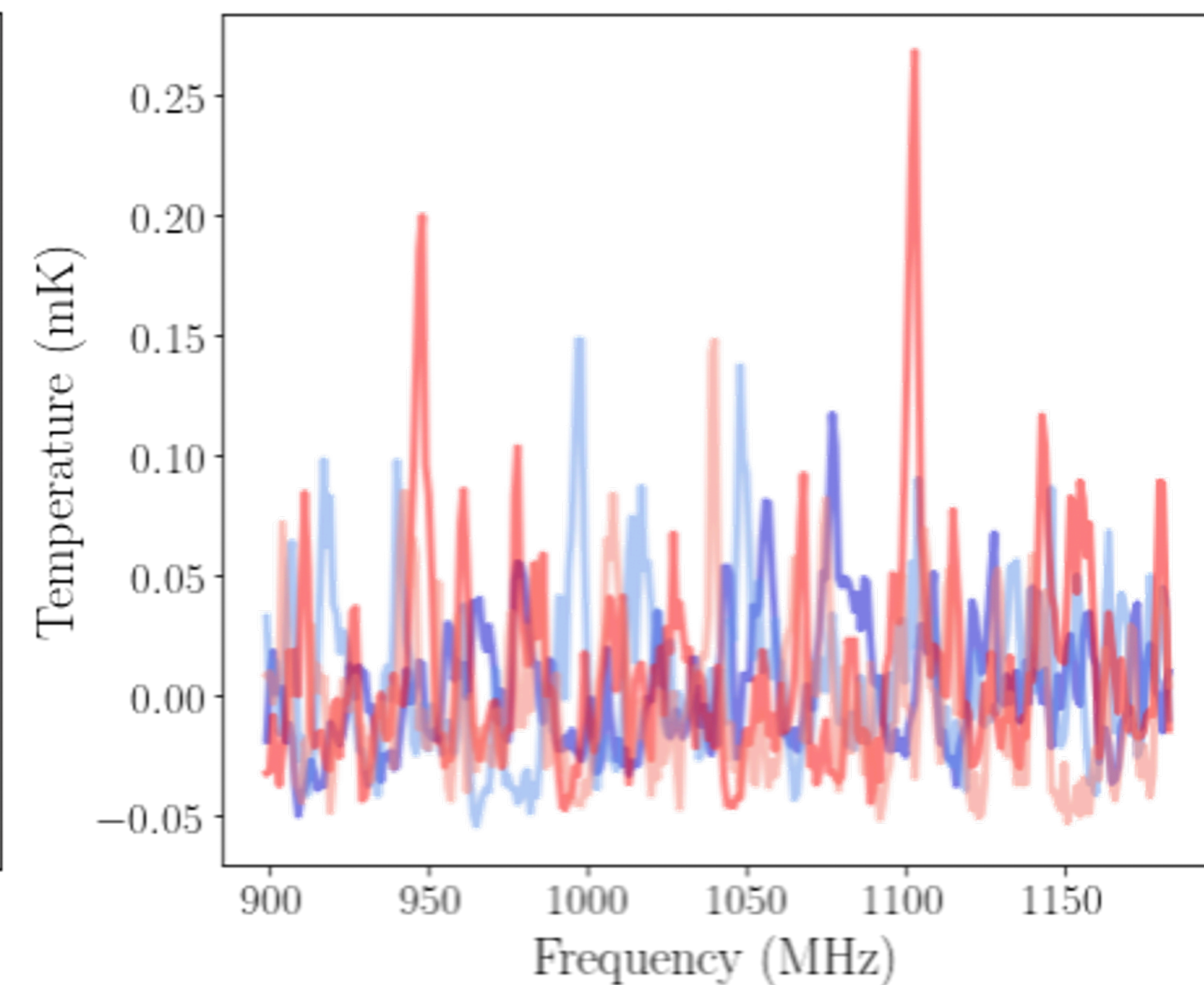
Polarised foregrounds K_{pol}

- Medium correlated (medium ℓ)
- Medium amplitude (medium σ^2)
- Overall smooth (large η)



21cm signal K_{21}

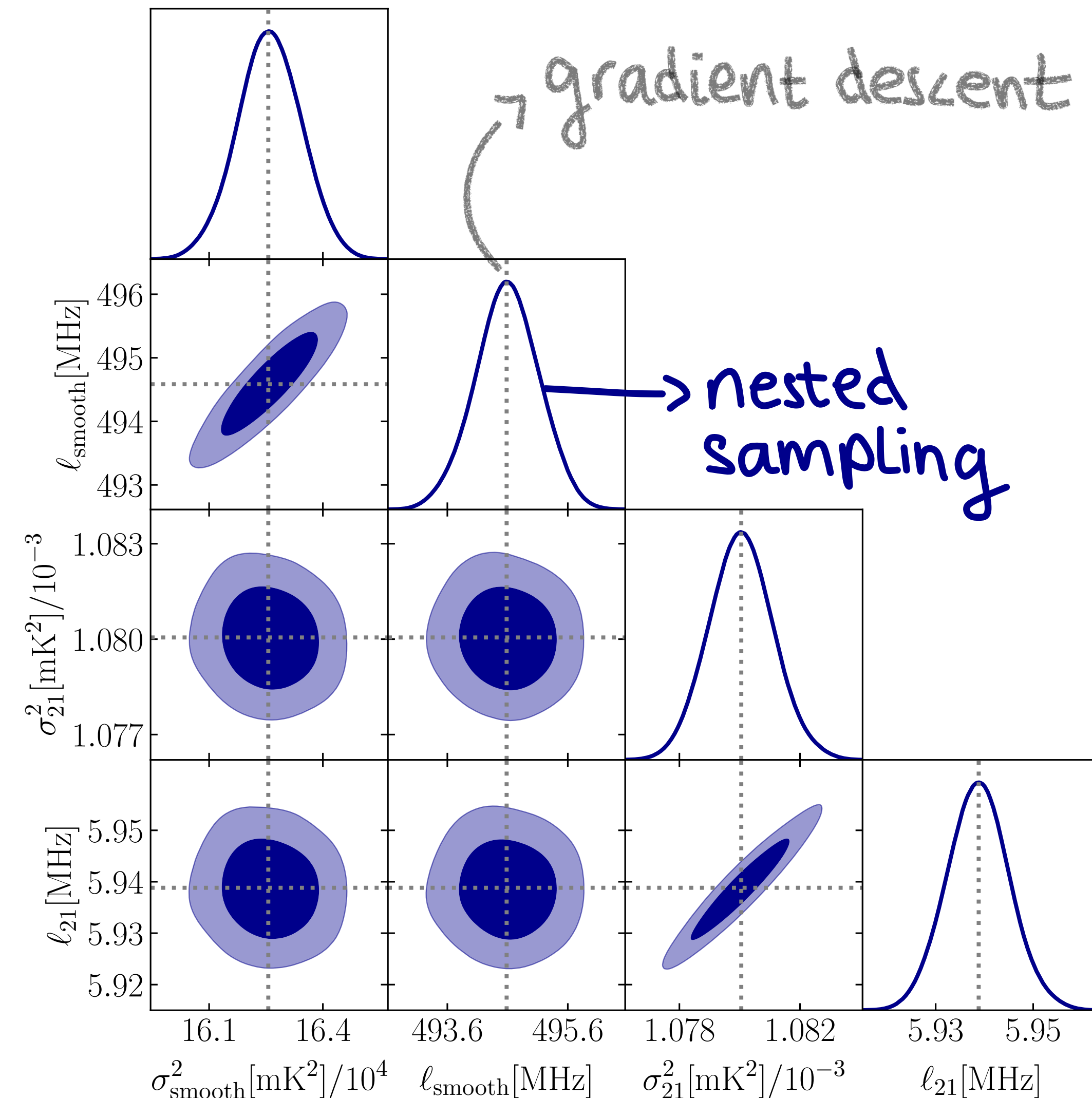
- Not correlated (small ℓ)
- Small amplitude (small σ^2)
- Not smooth (small η)



Finding the best hyperparameters

Finding the best-fitting covariance function K given our data

- We assume our data is Gaussian, so we can calculate the **marginal likelihood** analytically (fast), and find the hyperparameters ℓ and σ^2 that maximise it (e.g. *gradient descent*)
 - Do this for different choices of η , and compare the evidence to find the best choice
- Also can use *nested sampling*: more robust estimate of the evidence, and yields posterior distributions



Optimised covariance function

The best-fitting covariance function K given our data

21cm signal K_{21}

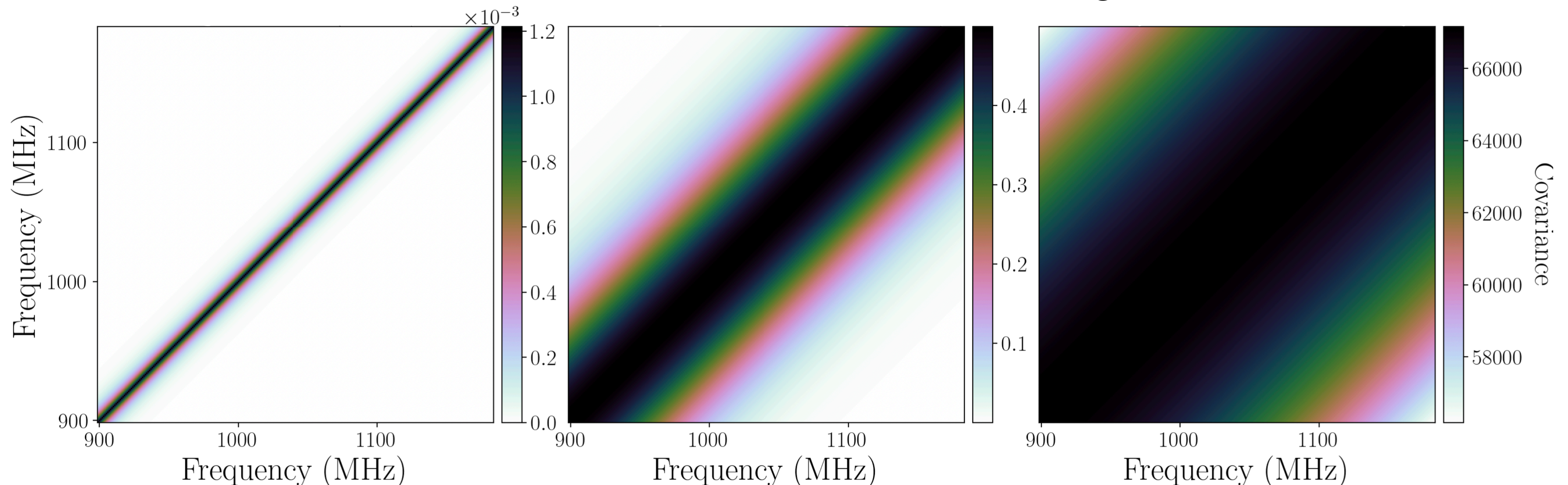
- Exponential function ($\eta = \frac{1}{2}$)
- Small lengthscale ℓ
- Small variance σ^2

Polarised foreground K_{pol}

- Radial basis function ($\eta \rightarrow \infty$)
- Medium lengthscale ℓ
- Medium variance σ^2

Smooth foreground K_{smooth}

- Radial basis function ($\eta \rightarrow \infty$)
- Large lengthscale ℓ
- Large variance σ^2



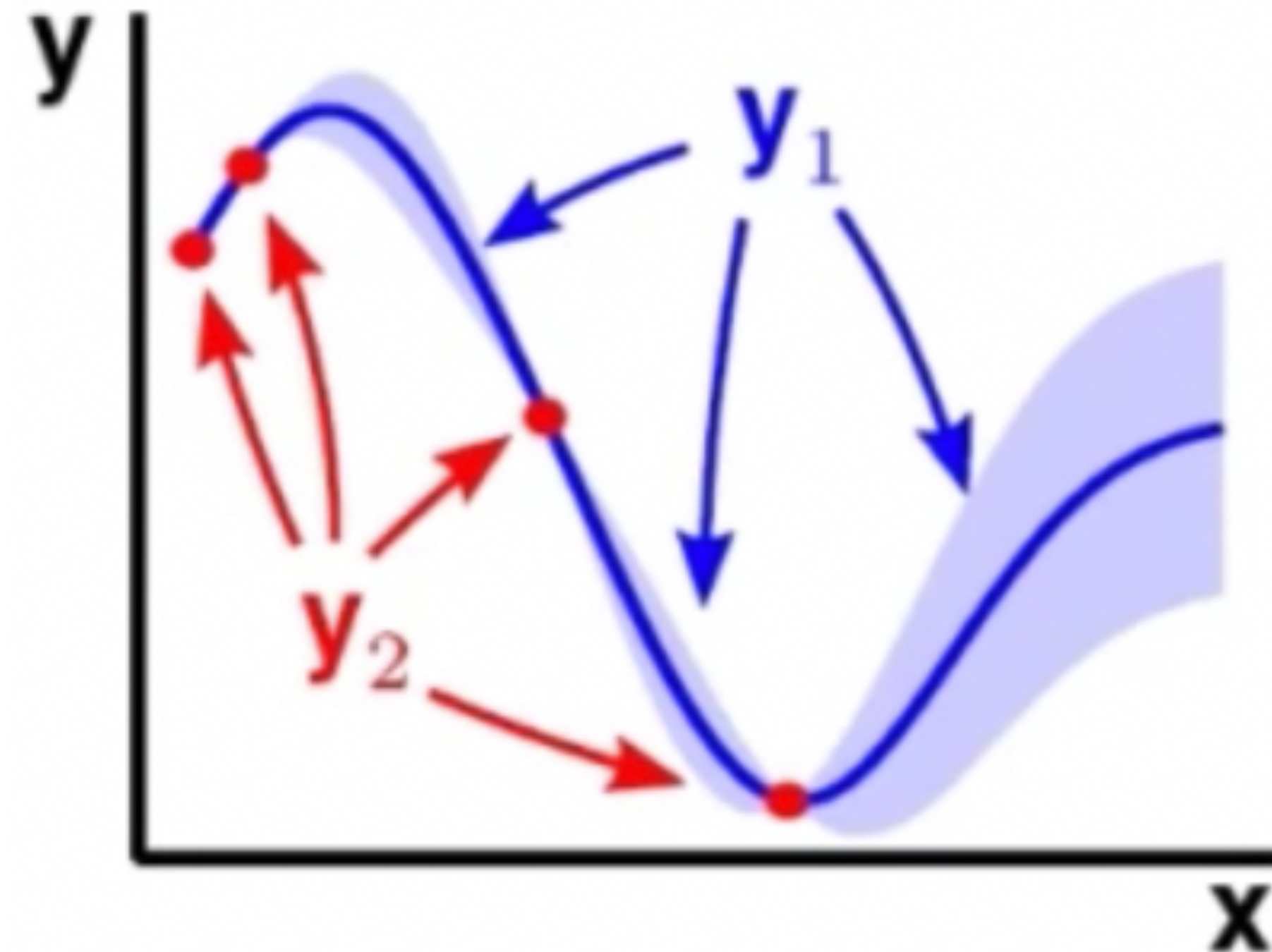
Foreground removal

How does **GPR** remove foregrounds? By predicting them!

Now we have: our data (d), its mean function (zero) and its best fitting covariance function ($K = K_{\text{fg}} + K_{21} + K_{\text{noise}}$). We can use this to *predict what the foregrounds look like in our frequency range*:

$$\begin{bmatrix} d \\ f_{\text{fg}} \end{bmatrix} = \mathcal{N} \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{\text{fg}} + K_{21} + K_{\text{noise}} & K_{\text{fg}} \\ \underbrace{K_{\text{fg}}}_{\text{Foreground covariance function}} & K_{\text{fg}} \end{bmatrix} \right)$$

Foreground
covariance function



Our foreground removal pipeline

How to remove foregrounds with GPR

1. Assume your data can be described as a Gaussian process, with covariance function: $K = K_{\text{fg}} + K_{21} + K_{\text{noise}}$
2. **Find the best-fitting covariance function K using e.g. nested sampling**
3. Use your data and its covariance function to predict the foregrounds
4. Remove foreground prediction from your data!

→ Hardest part

If you're interested in running this pipeline...


Our code `gpr4im` is available at:
github.com/paulassoares/gpr4im

Easy to install:
`pip install gpr4im`

Introductory notebooks that run through the pipeline step-by-step

main 1 branch 0 tags

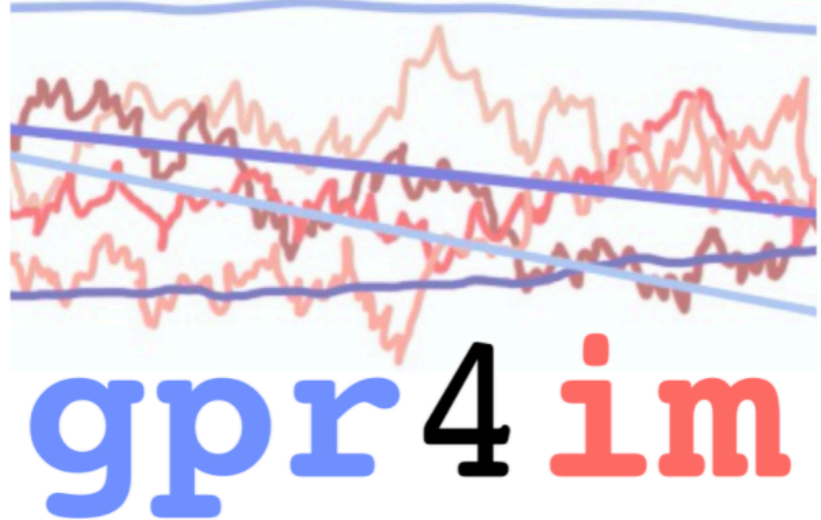
Go to file Add file Code

 paulassoares aligned logo to center

c1ea849 27 seconds ago 80 commits

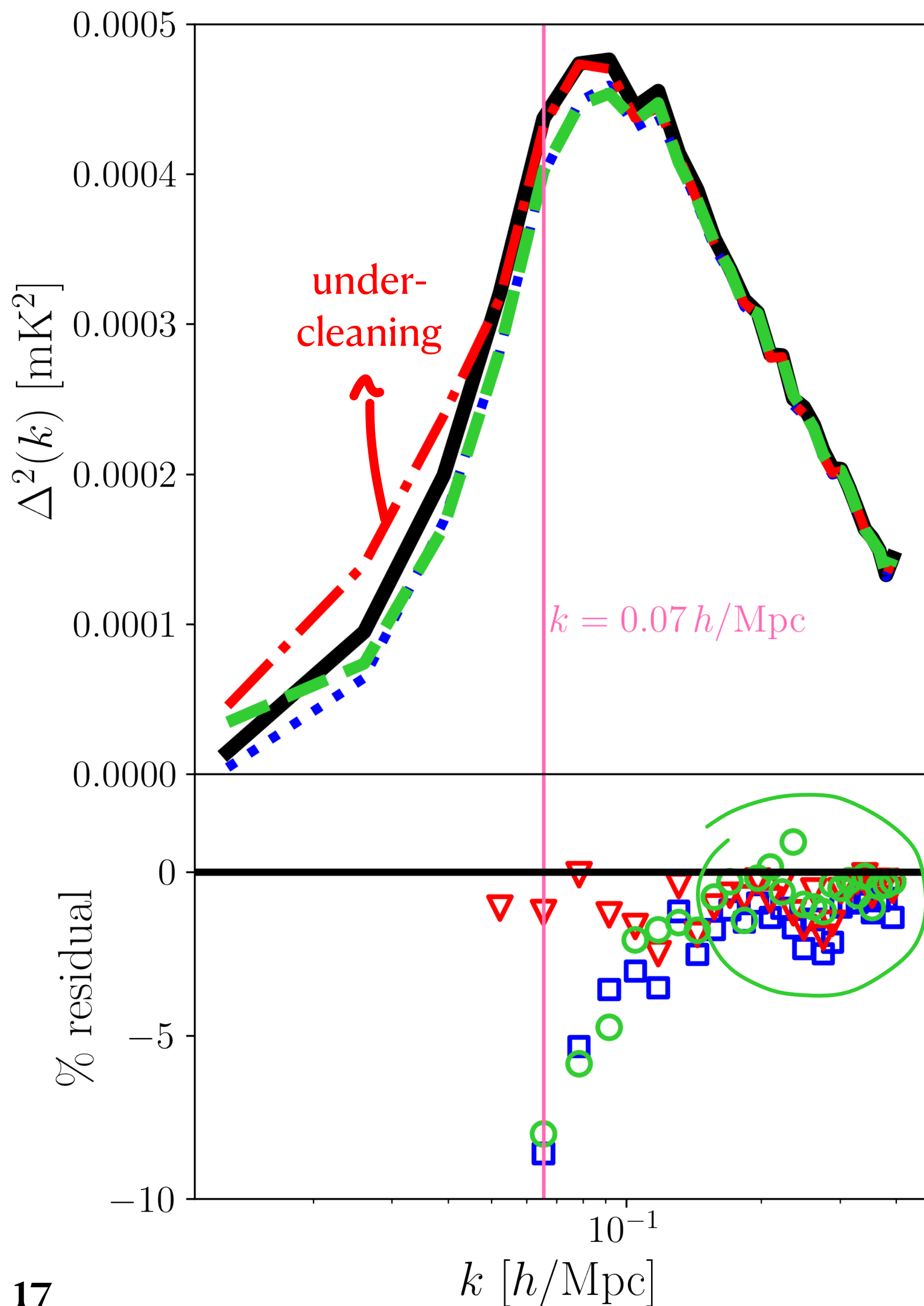
Data	fixed typos in readmes and jupyter notebook descriptions	15 days ago
Jupyter Notebooks	added logo file	9 minutes ago
gpr4im	fixed pypi readme	2 days ago
tests	updated data format, added analysis notebooks	2 months ago
LICENSE	create LICENSE	last month
README.md	aligned logo to center	27 seconds ago
setup.py	fixed pypi readme	2 days ago

README.md



gpr4im

This package uses Gaussian Process Regression (GPR) as a foreground removal technique in the context single

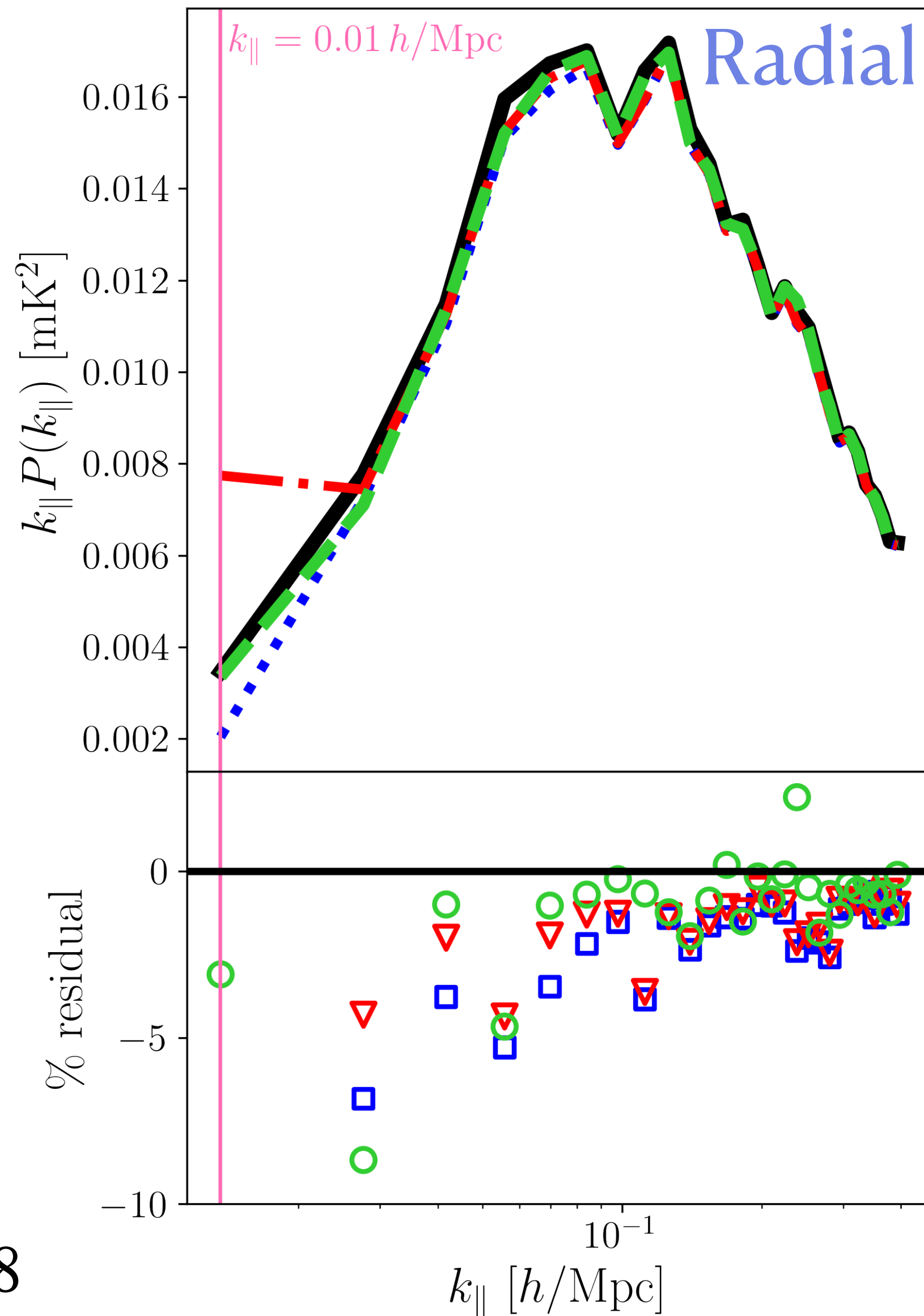


Results

No polarisation

- True HI power spectrum is the black solid line, *what we want to recover*
- **GPR** results are in **green**
- PCA results are in **red** ($N_{\text{fg}} = 2$) and **blue** ($N_{\text{fg}} = 3$)
- Bottom panel shows percentage residual difference from truth
- **Pink** line shows k -bin below which **GPR** diverges above 10% from the truth

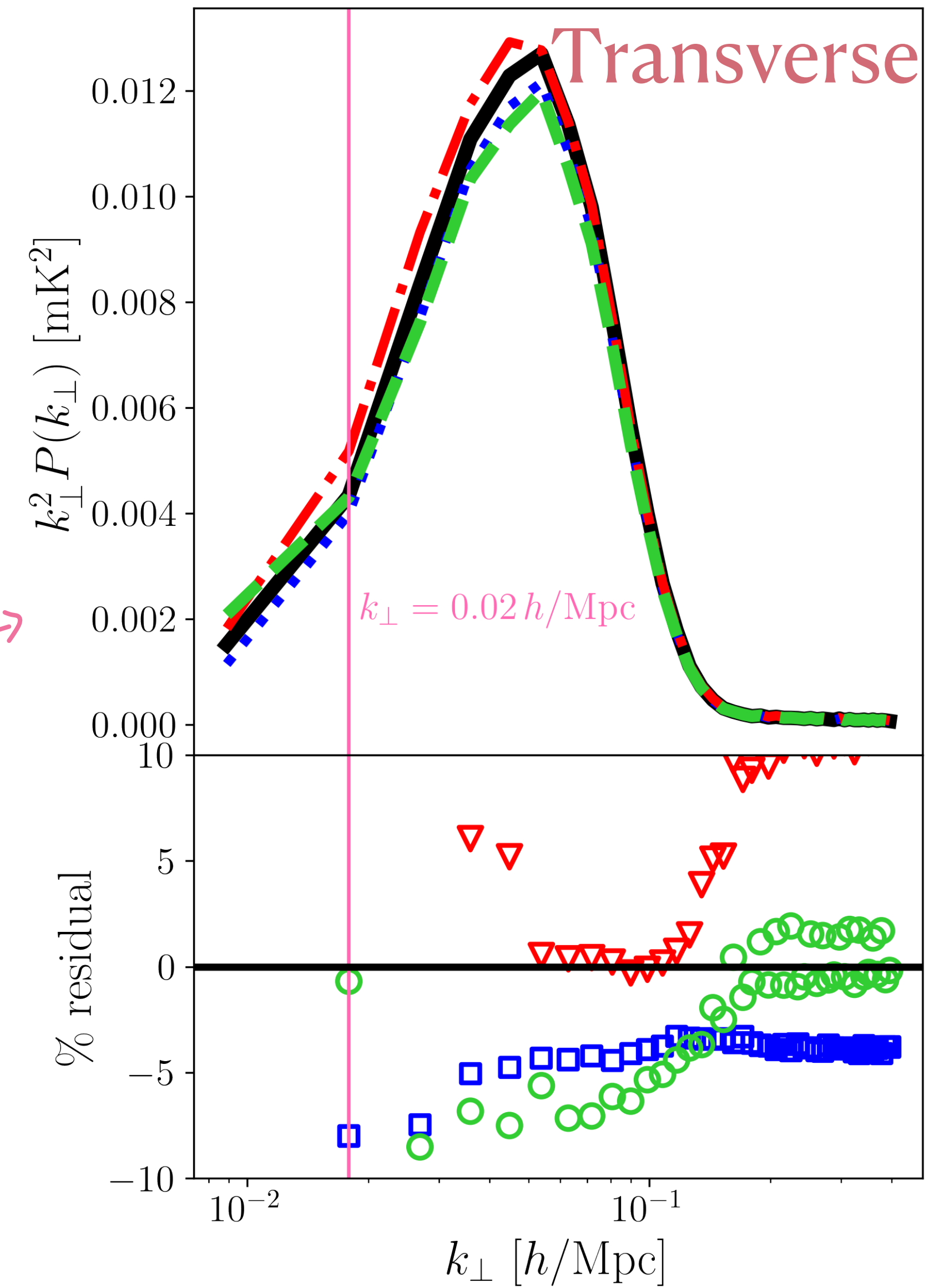
Results



No polarisation

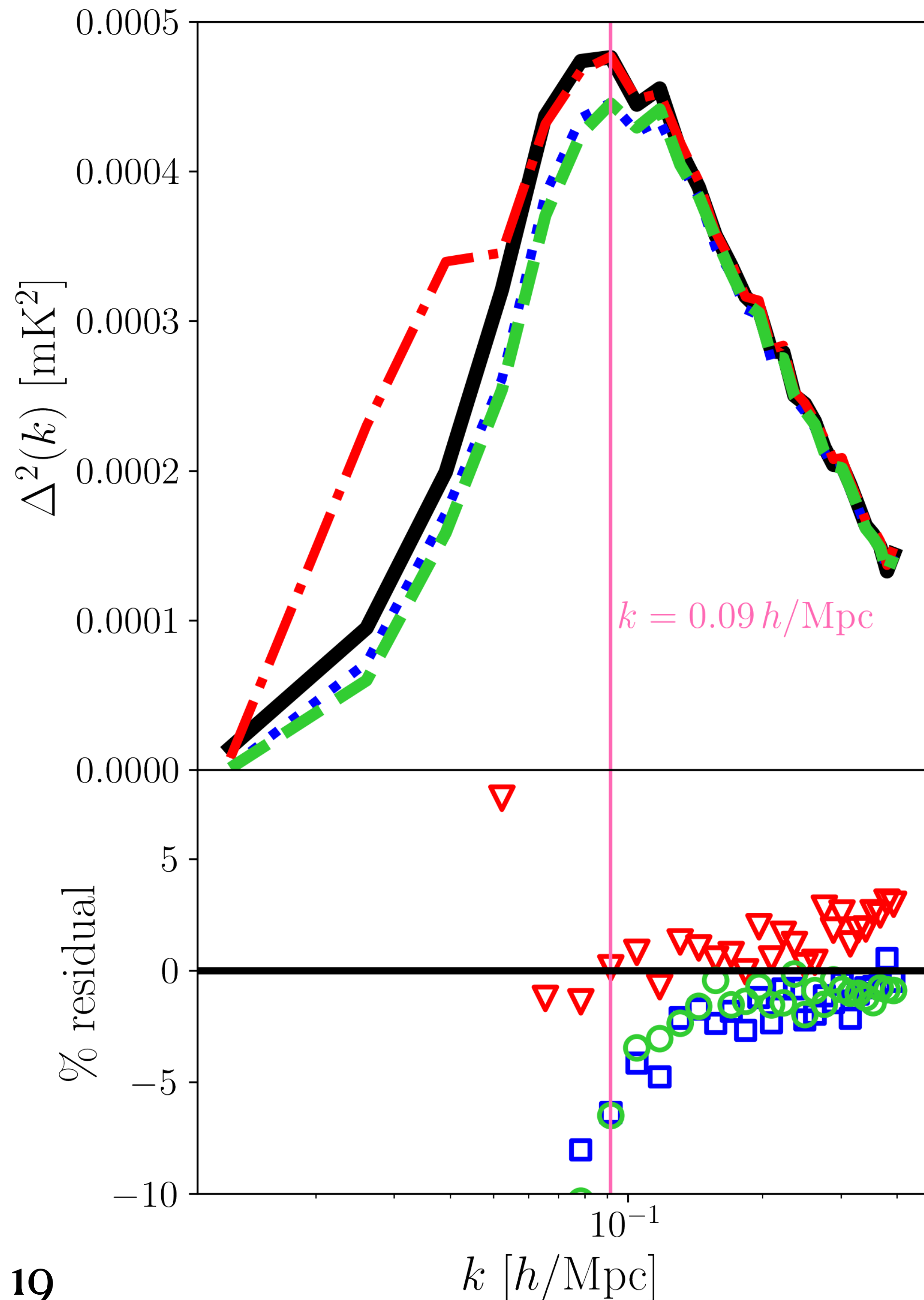
- Very good
- **GPR** is better than PCA on all scales
- **GPR** recovers the full range of the radial power spectrum within 10% residual
- Less good
- **GPR** better on small scales where beam dominates
- **GPR** cannot recover full range of transverse power spectrum within 10% residual

GPR is better in the radial direction

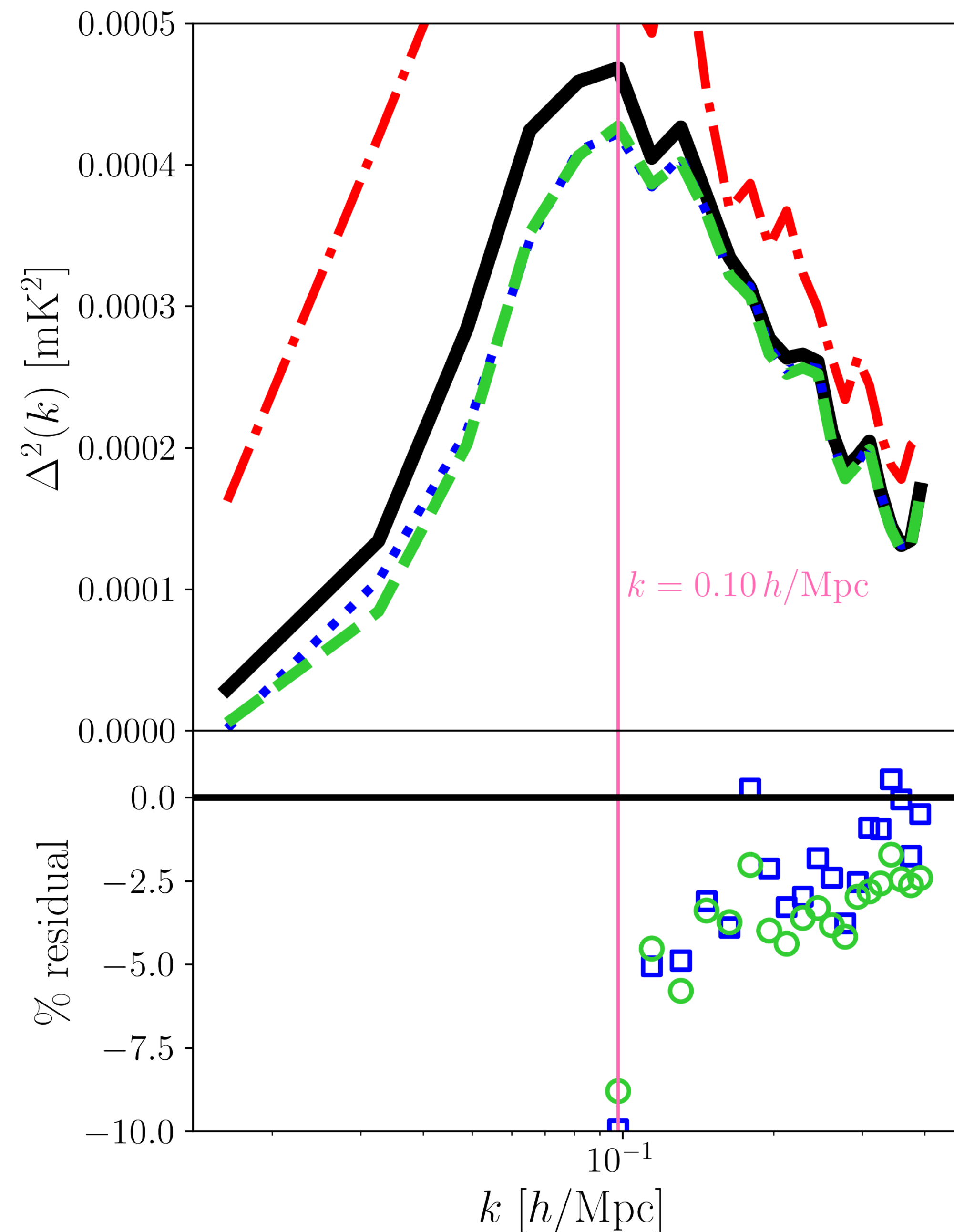


Results

With polarisation



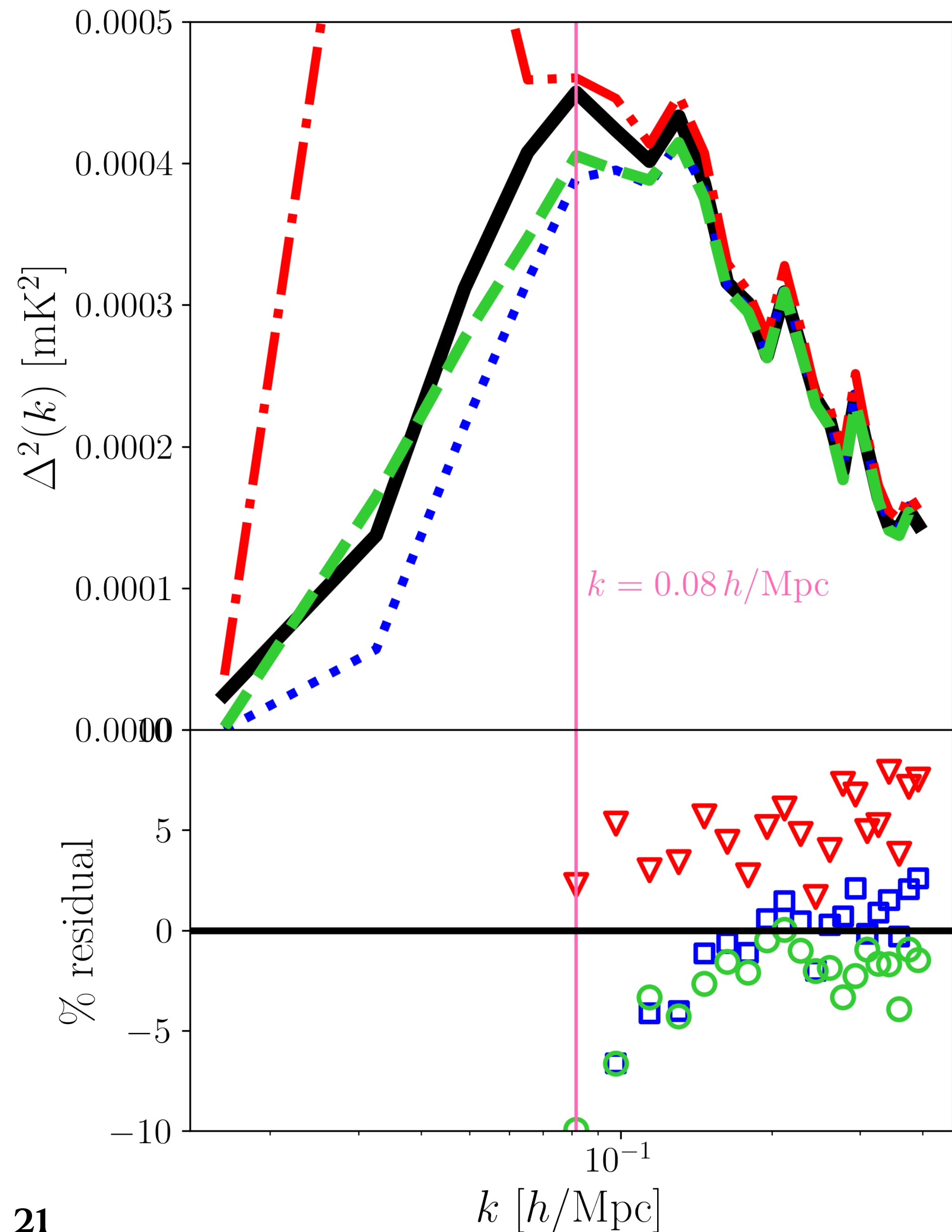
- **GPR** results are in **green**
- PCA results are in **red** ($N_{\text{fg}} = 6$) and **blue** ($N_{\text{fg}} = 7$)
- **GPR** performs *worse* in the presence of polarised foregrounds
- Somewhat better than PCA on small scales, but PCA ($N_{\text{fg}} = 7$) can recover larger scales



Bandwidth/redshift dependence

High frequency, low redshift

- **GPR** results are in **green**
- PCA results are in **red** ($N_{\text{fg}} = 3$) and **blue** ($N_{\text{fg}} = 4$)
- In this case, **GPR** performs worse than PCA ($N_{\text{fg}} = 4$), and worse than in the *full bandwidth* case



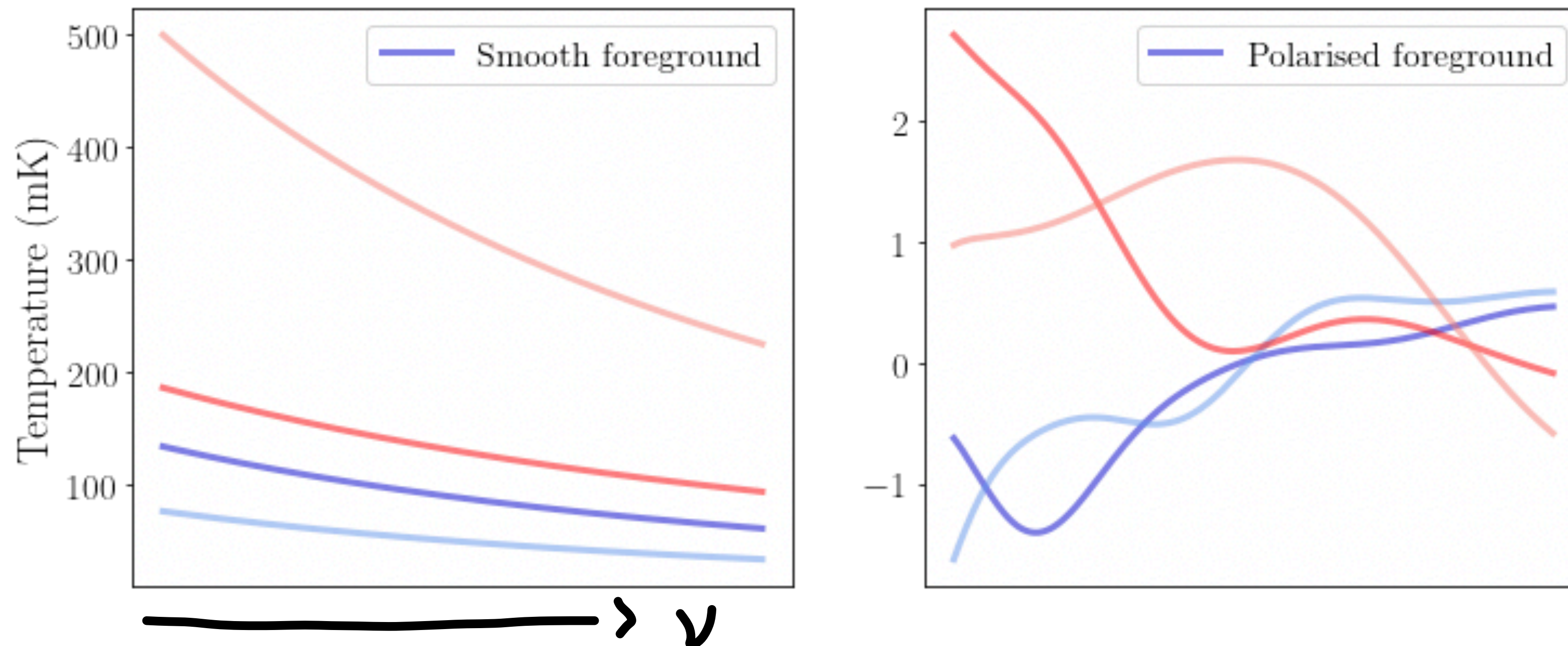
Bandwidth/redshift dependence

Low frequency, high redshift

- **GPR** results are in **green**
- PCA results are in **red** ($N_{\text{fg}} = 4$) and **blue** ($N_{\text{fg}} = 5$)
- In this case, both PCA cases lead to under-cleaning, but **GPR** only over-cleans, and can access larger scales, so *GPR performs better*
- It also works better than in the *full bandwidth* case

Bandwidth/redshift dependence

- Is half bandwidth better than full bandwidth? e.g. [Hothi et al. \(2020\)](#)
 - Unclear: The low redshift case is *worse* than the full bandwidth, but the high redshift is *better*
- Interesting that the high redshift (brighter foregrounds) case is better



Key takeaways

- *It is possible to run GPR for foreground removal technique in the case of single-dish, low redshift HI intensity mapping*
- GPR performs better in the radial direction than in the transverse direction
- GPR performs better than PCA in the no polarisation case, and similar when including polarisation
 - Polarisation leakage makes GPR foreground removal more difficult
- GPR performs better at high redshifts than low redshifts
- For PCA, we constantly needed to change N_{fg} depending on bandwidth size, missing channels, including polarisation, etc.
 - **GPR does not require this fine tuning, it finds the best fitting covariance model given the data**
- Our code is available on github.com/paulassoares/gpr4im