Numerical generation of fractal surfaces:

a flexible and controllable approach

A TribNet Webinar Presented By

INVITED SPEAKERS

PROF. ANDREAS ALMQVIST
LULEÅ UNIVERSITY OF TECHNOLOGY
DR. FRANCESC PEREZ-RÁFOLS
UNIVERSITAT POLITÈCNICA DE CATALUNYA

30 JUNE 2023

EPSEM – Escola Tècnica Superior d'Enginyeria de Manresa

LULEÅ UNIVERSITY OF TECHNOLOGY

Outline

- Motivation
  - Why do we need generated surfaces
  - Height probability function
  - Power Spectrum
- Algorithm
  - Description
  - A note of caution
- Implementation example

Where to find the information

- Article

  F. Pérez-Ràfols, A. Almqvist, *Generating randomly rough surfaces with given height probability distribution and power spectrum*, Tribology International, Volume 131, pp. 591-604 (2019)

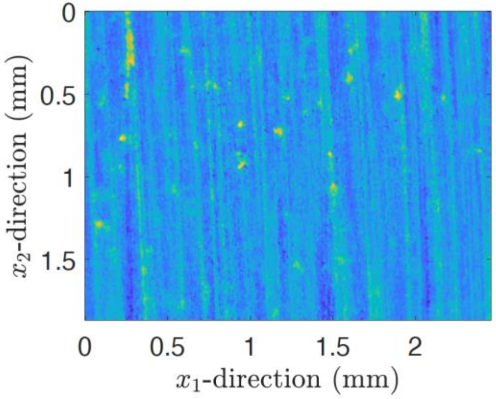
- Code

  https://www.mathworks.com/matlabcentral/fileexchange/129469-fractal-surface-generator

Motivation – Measured vs. Generated

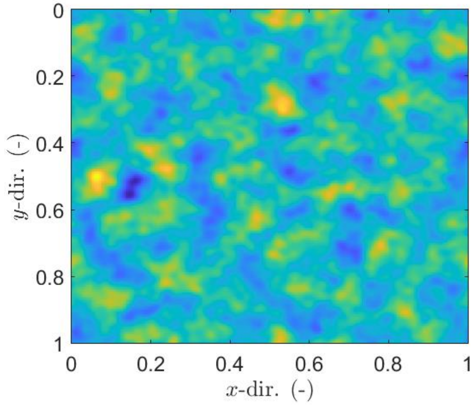
Measured surfaces

- Close to reality
- Uncontrollable features
- High uncertainty
- Expensive to obtain



Generated Surface

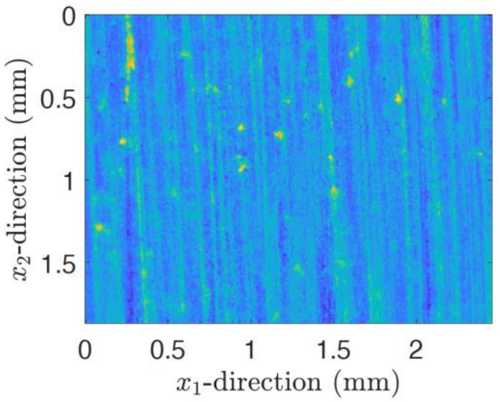
- Idealization
- Controlled if done right
- Moderate uncertainty
- Readily available



Motivation – Measured vs. Generated

Measured surfaces

- Close to reality
- Uncontrollable features
- High uncertainty
- Expensive to obtain



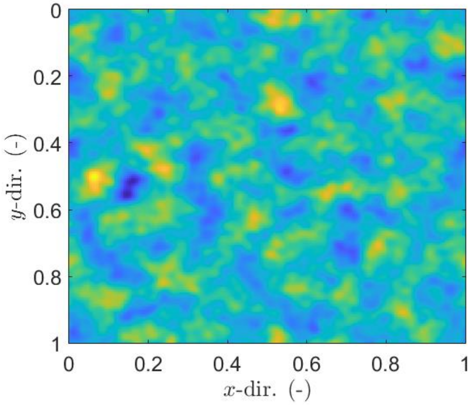
It is hard to do parametric studies with measured surfaces

It is more fruitful to develop the theory on generated surfaces and then test it on measured ones

We want the generated surfaces to be as close as possible to the real ones

Generated Surface

- Idealization
- Controlled if done right
- Moderate uncertainty
- Readily available



Motivation – Surface Roughness Parameters

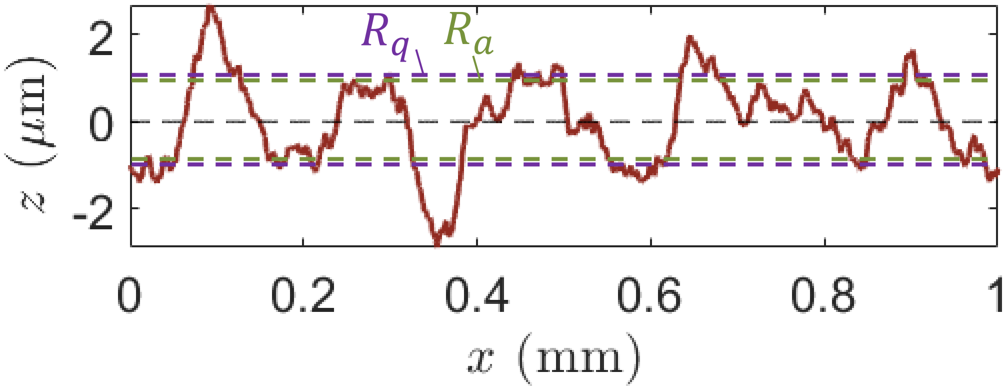
Average height

RMS roughness:

$$R_q = \sqrt{\frac{1}{N}\sum(z-\bar{z})^2}$$

Mean absolute value:

$$R_a = \frac{1}{N}\sum|z-\bar{z}|$$

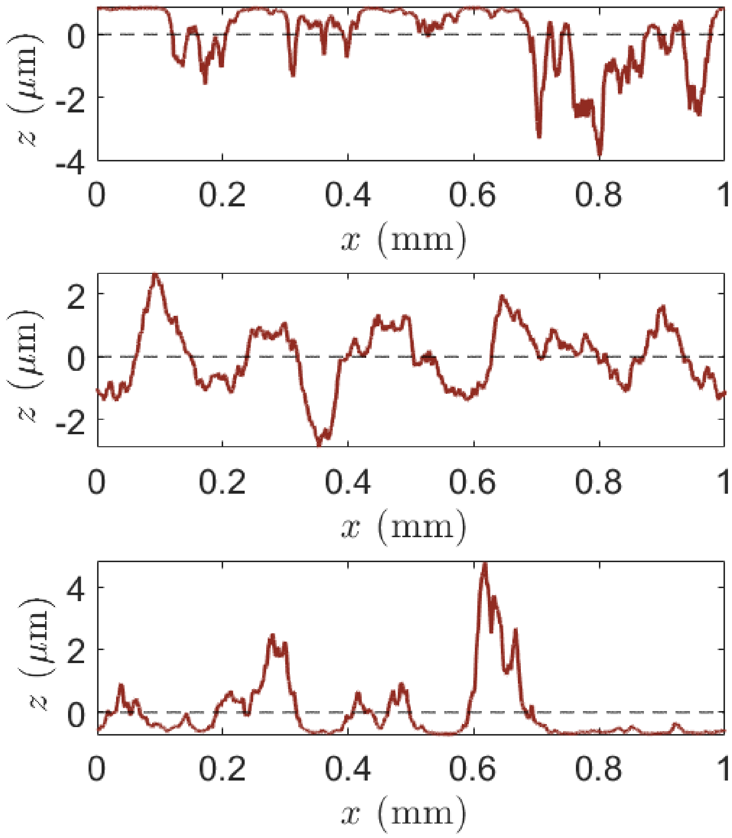


Both are quite similar and give a sense of how large roughness is

Motivation – Surface Roughness Parameters

Other height parameters

The three profiles have the same R_q, but they are extremely different



Other height parameters

The three profiles have the same R_q, but they are extremely different

Skewness:

$$R_{sk} = \frac{1}{R_q^3}\left(\frac{1}{N}\sum(z-\bar{z})^3\right)$$

R_{sk} characterises symmetry of the HPD

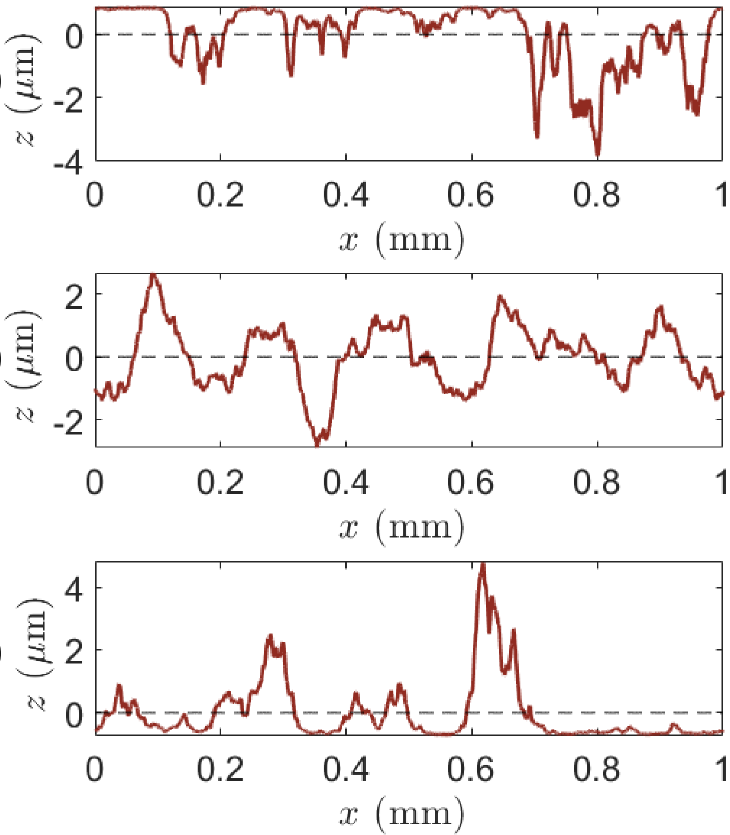
$R_{sk} > 0$: more peaks than valleys

$R_{sk} < 0$: more valleys than peaks

$\boldsymbol{R_{sk} < 0}$

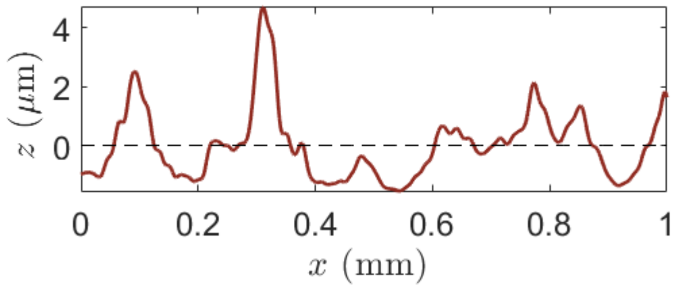
$\boldsymbol{R_{sk} = 0}$

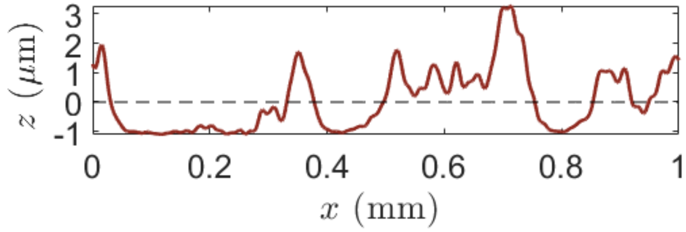
$\boldsymbol{R_{sk} > 0}$



Other height parameters

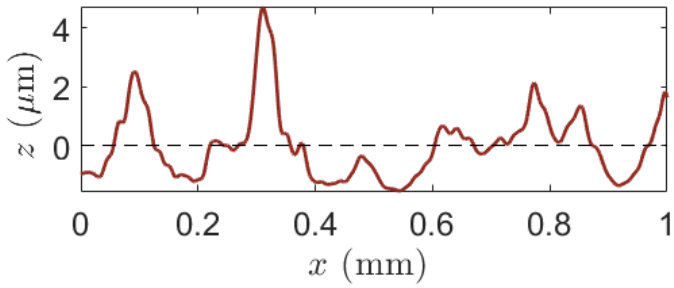
This two profiles have the same R_q and R_{sk}, but they are still quite different





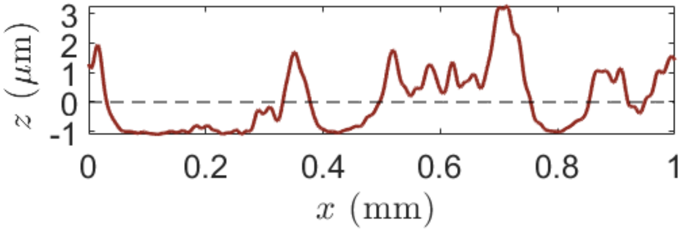
Other height parameters

This two profiles have the same R_q and R_sk, but they are still quite different



Higher order moments:

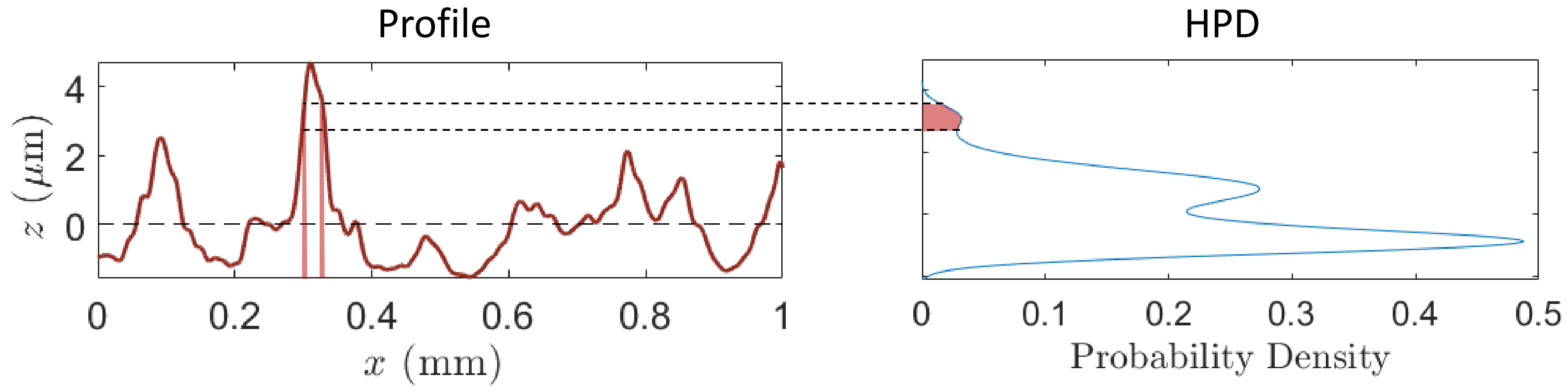
$$R_{ku} = \frac{1}{R_q^4}\left(\frac{1}{N}\sum(z-\bar{z})^4\right)$$



$$m_5 = \frac{1}{R_q^5}\left(\frac{1}{N}\sum(z-\bar{z})^5\right)$$

…

Motivation – Height Probability Density (HPD)

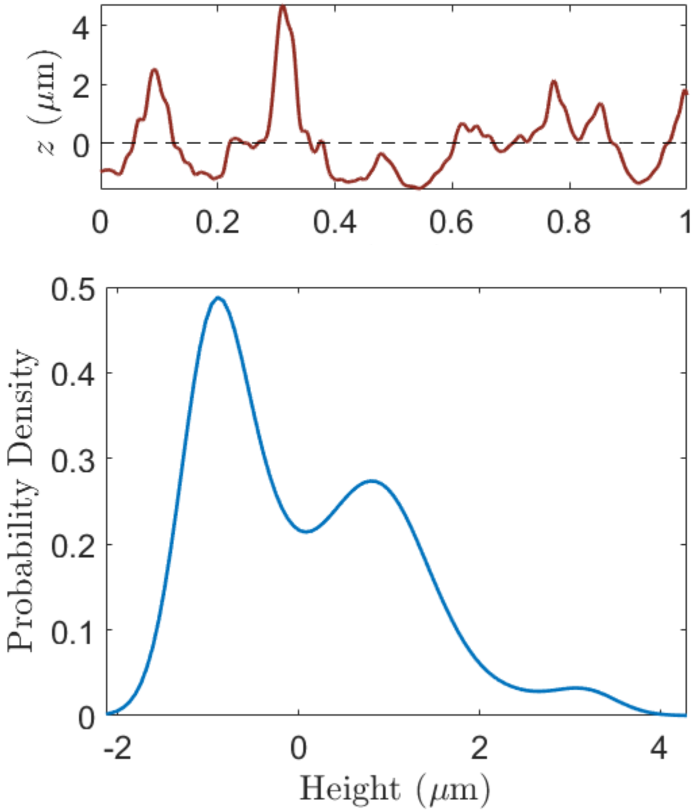


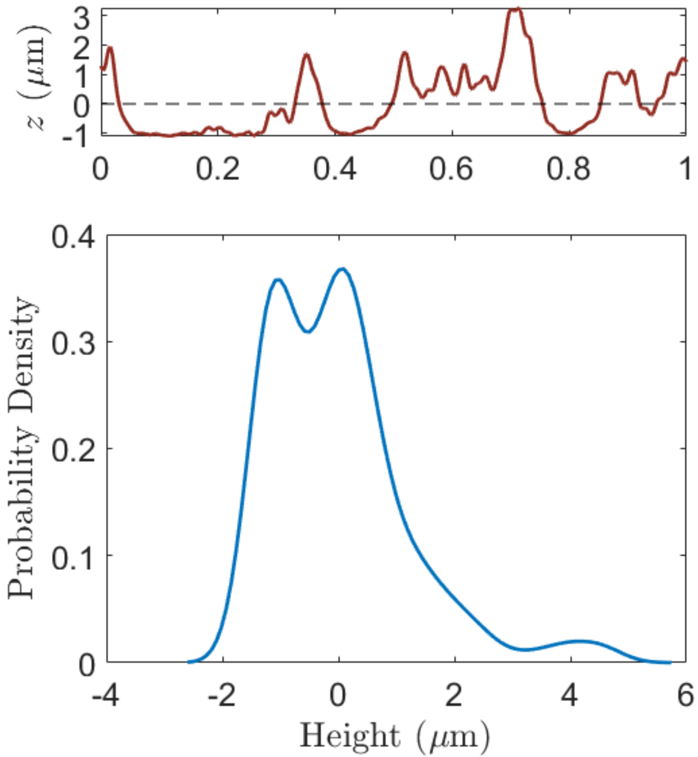
The area indicates the probability that a given point lays between two heights

It can also be thought of as the fraction of points within that height range

It contains all information on height; height parameters summarize its information

Motivation – Height Probability Density (HPD)





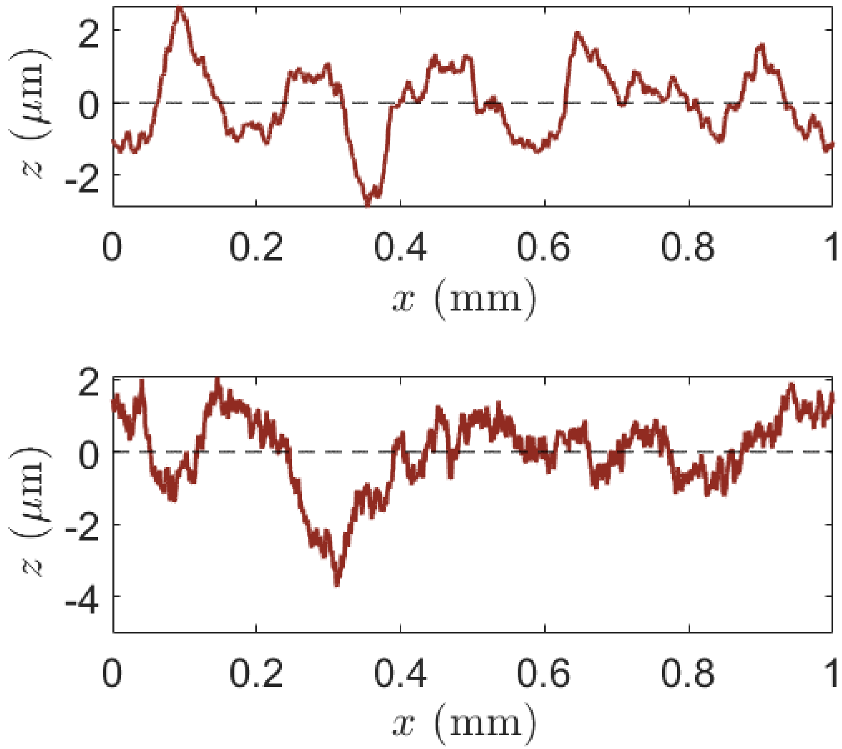
Motivation – Power Spectral Density (PSD)

The two profiles have the **same HPD**

Nonetheless, we see that they are **notably different**

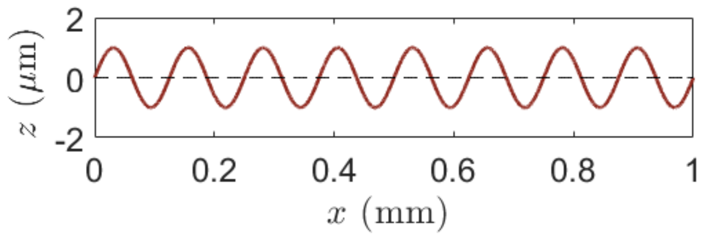
Controlling height is not enough to specify a surface

Lateral information is also needed

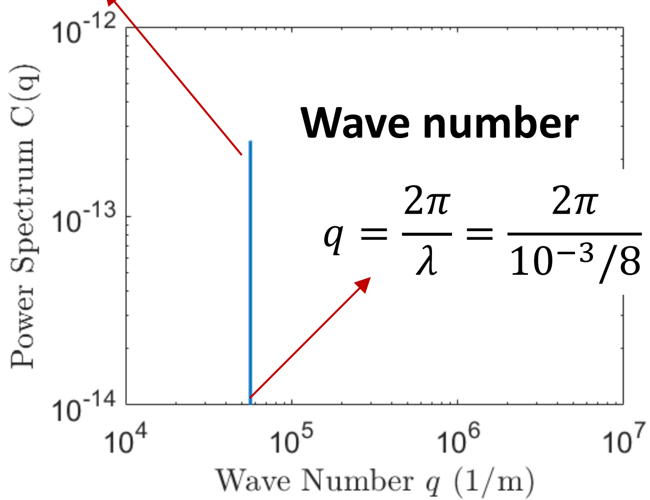


Motivation – Power Spectral Density (PSD)

The Power Spectral Density (PSD) indicates the amplitude of a wave with a given wavelength.



Amplitude $\Delta = (10^{-6})^2$



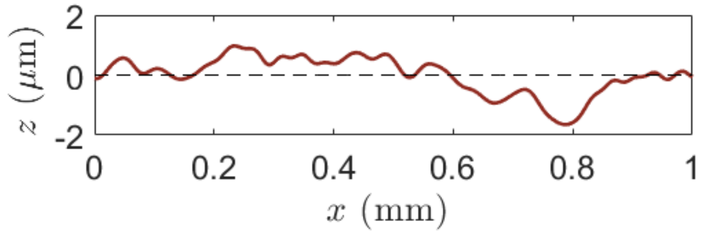
Wave number

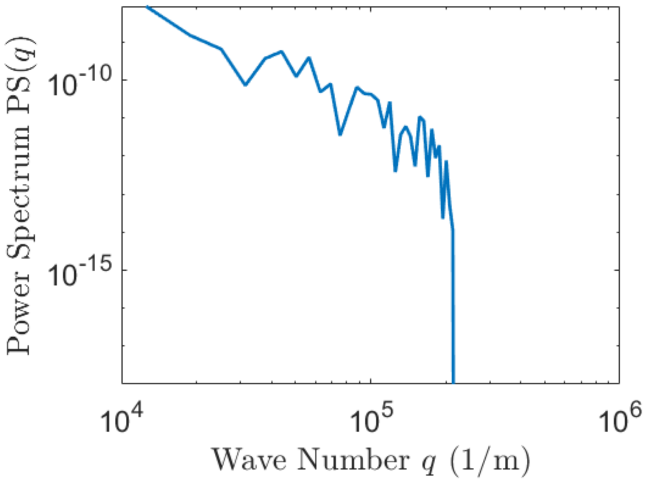
$$q = \frac{2\pi}{\lambda} = \frac{2\pi}{10^{-3}/8} = 5 \cdot 10^4$$

Motivation – Power Spectral Density (PSD)

The Power Spectral Density (PSD) indicates the amplitude of a wave with a given wavelength.

The PSD of z is given by $|\mathcal{F}\{z\}|^2$, the square of the amplitude Fourier transform.



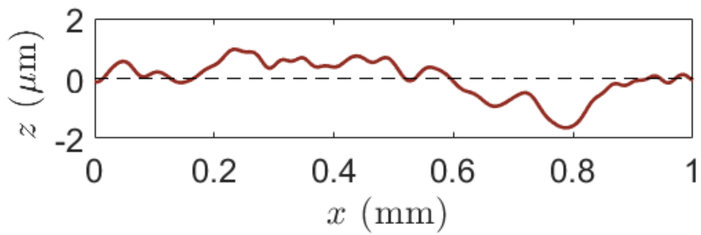


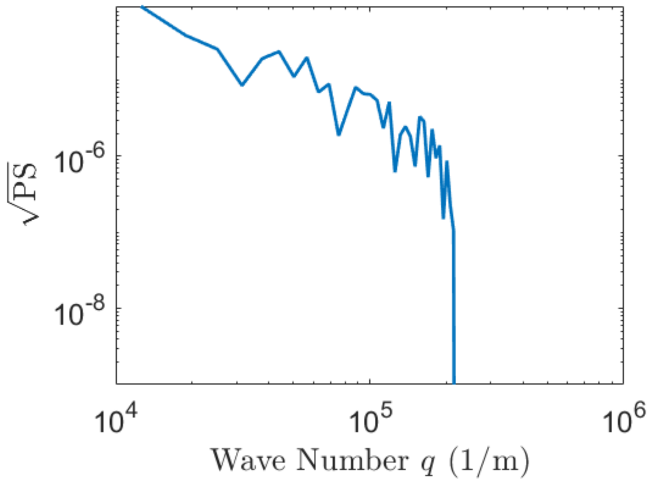
Motivation – Power Spectral Density (PSD)

The Power Spectral Density (PSD) indicates the amplitude of a wave with a given wavelength.

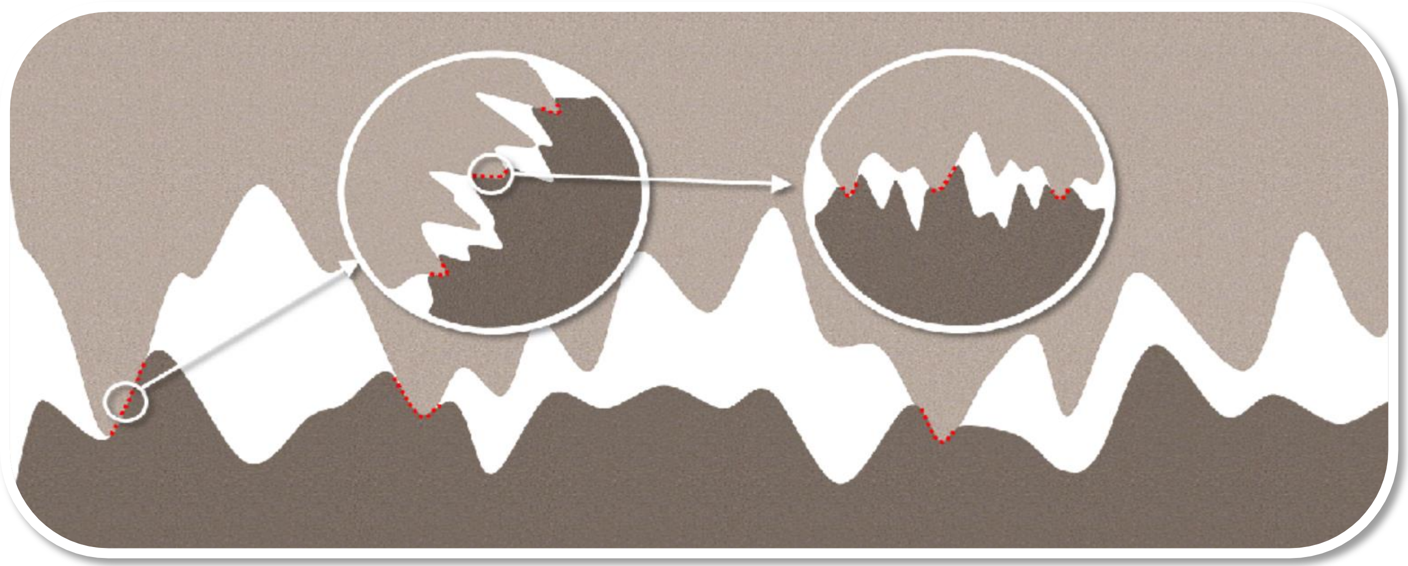
The PSD of z is given by $|\mathcal{F}\{z\}|^2$, the square of the amplitude Fourier transform.

For the algorithm it is convenient to work with $\sqrt{PSD} = |\mathcal{F}\{z\}|$





Motivation – Self Affine Surfaces



As we zoom in a surface, we see ever more rough details

In most cases, the smaller scales “look the same” as the larger ones

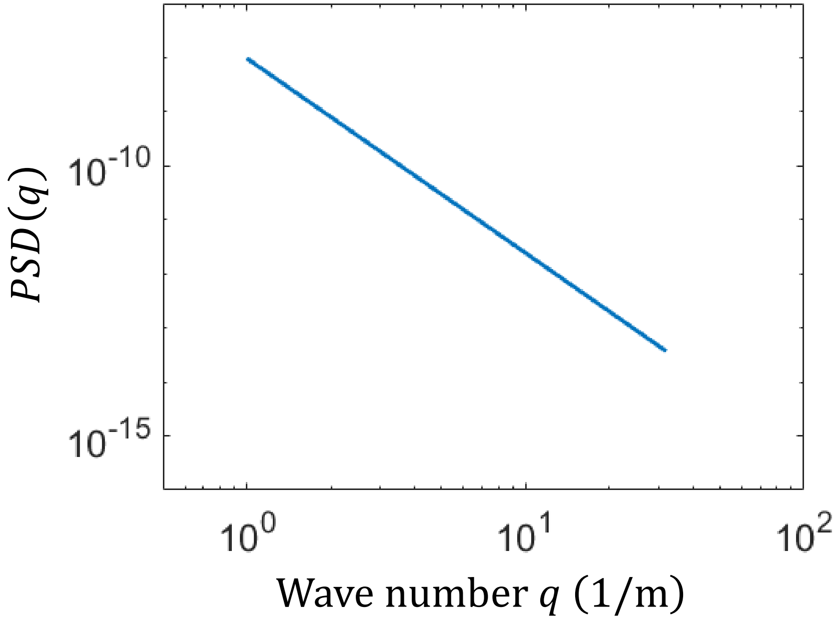
Motivation – Self Affine Surfaces

A common way to study these kind of surfaces is to define them as **Self Affine**

These have a PSD of the form

$$PSD(q) \propto q^{-2(1+H)},$$

where H is the Hurst exponent



Motivation – Self Affine Surfaces

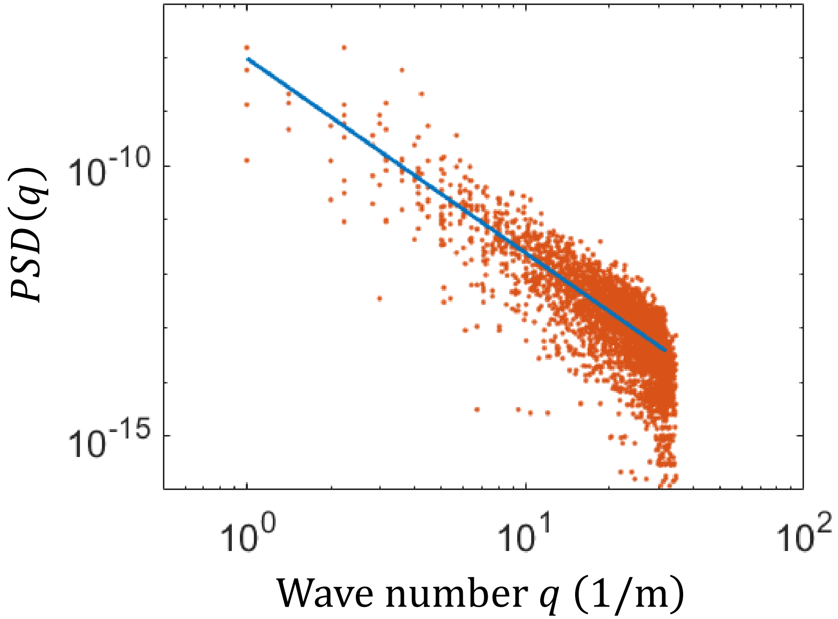
A common way to study these kind of surfaces is to define them as **Self Affine**

These have a PSD of the form

$$PSD(q) \propto q^{-2(1+H)},$$

where H is the Hurst exponent

It is readily understood in an averaged sense



Motivation – Self Affine Surfaces

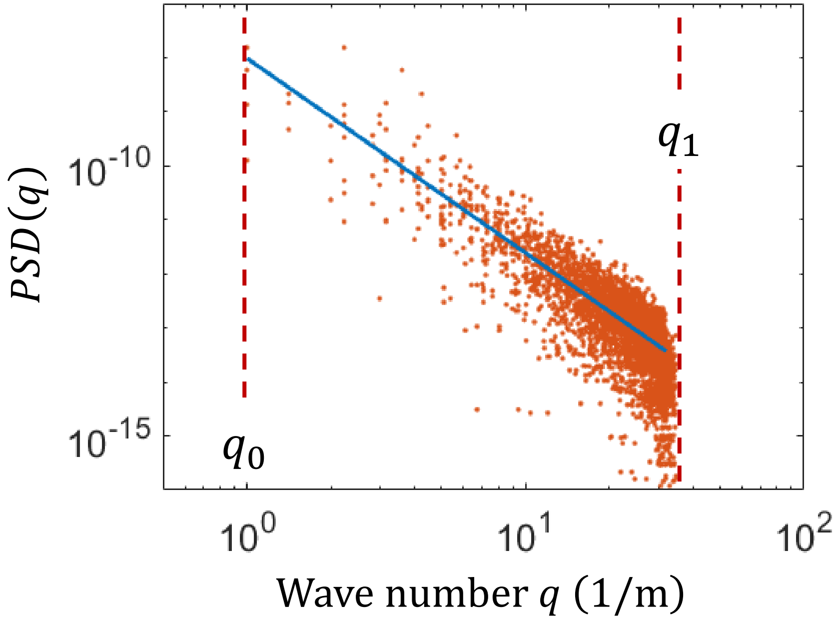
In practice, self affinity will only be considered in a limited region of wave numbers, i.e.,

$$PSD(q) = \begin{cases} 0 & q < q_0 \\ Cq^{-2(H+1)} & q_0 < q < q_1 \\ 0 & q_1 < q \end{cases}$$

where

$q_0 = \dfrac{2\pi}{\lambda_0}$ indicates longest wavelength present,

$q_1 = \dfrac{2\pi}{\lambda_1}$ indicates shortest wavelength present

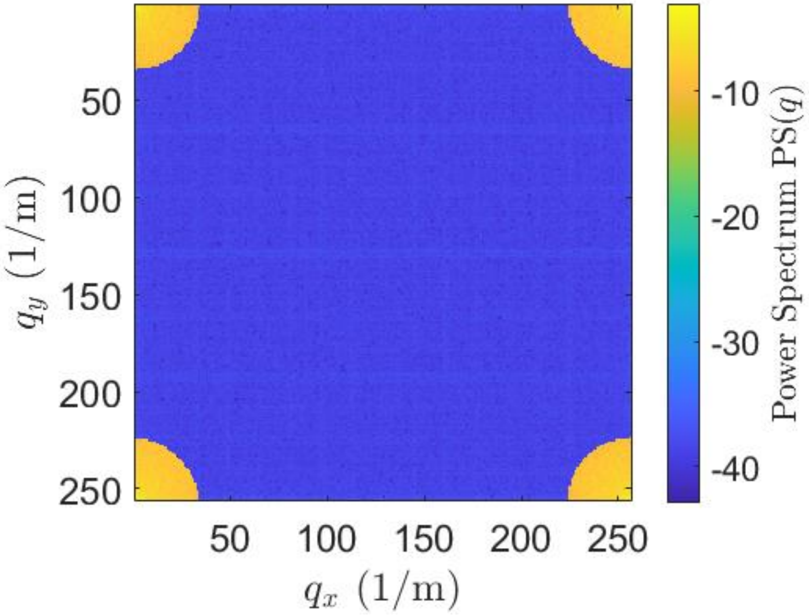


Motivation – Self Affine Surfaces

In surfaces we have

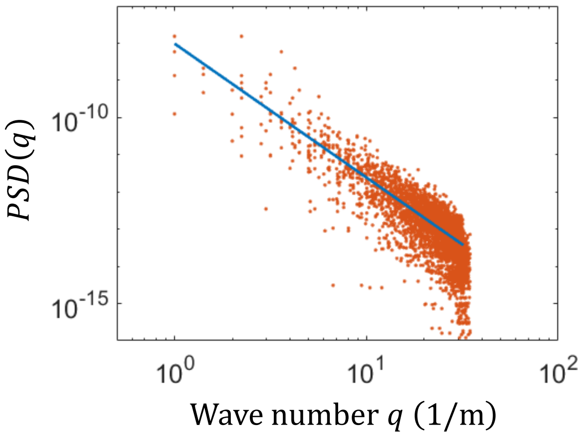
$$\vec{q} = [q_x, q_y]$$

$$q = \sqrt{q_x^2 + q_y^2}$$

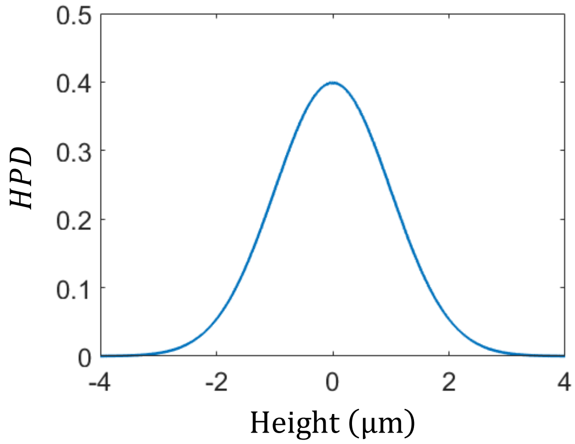


Motivation – Conventional Surface Generation

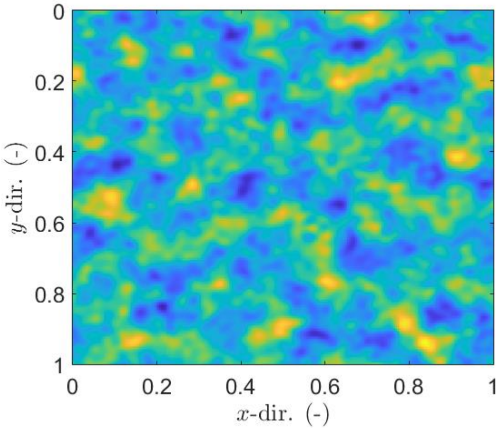
Self-affine PSD



Gaussian HPD



Resulting surface

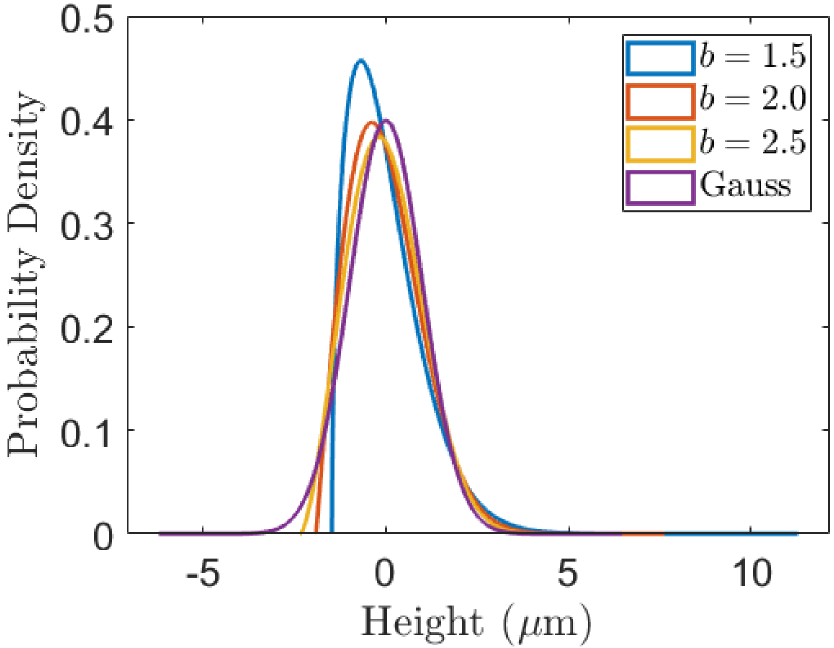


Motivation – Non-Gaussian Surfaces

Weibull HPD

$$f_W(z) = \begin{cases} \dfrac{b}{-(z-z_0)}\left(\dfrac{-(z-z_0)}{a}\right)^{b-1} & (z-z_0 \leq 0) \\ 0 & \text{otherwise} \end{cases}$$

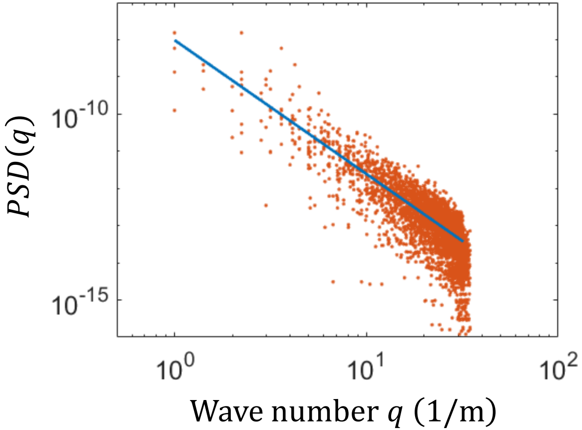
a: size parameter (controls $\mathrm{R_q}$)
b: shape parameter



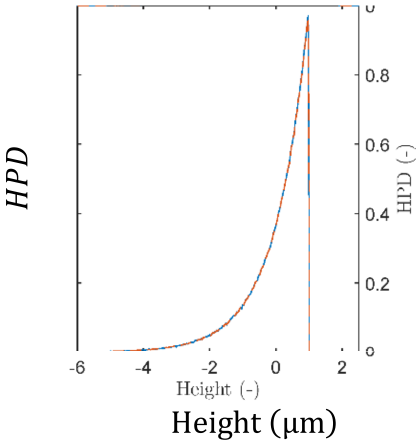
Motivation – Non-Gaussian Surfaces

Weibull HPD

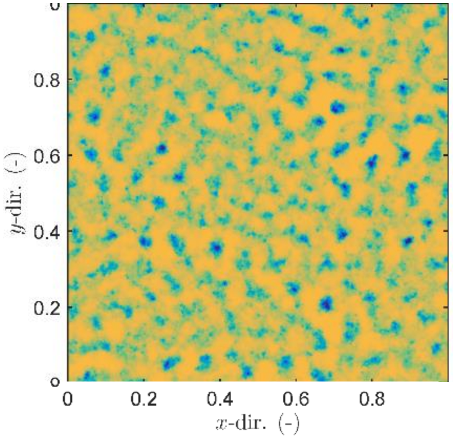
Self-affine PSD



Weibull HPD

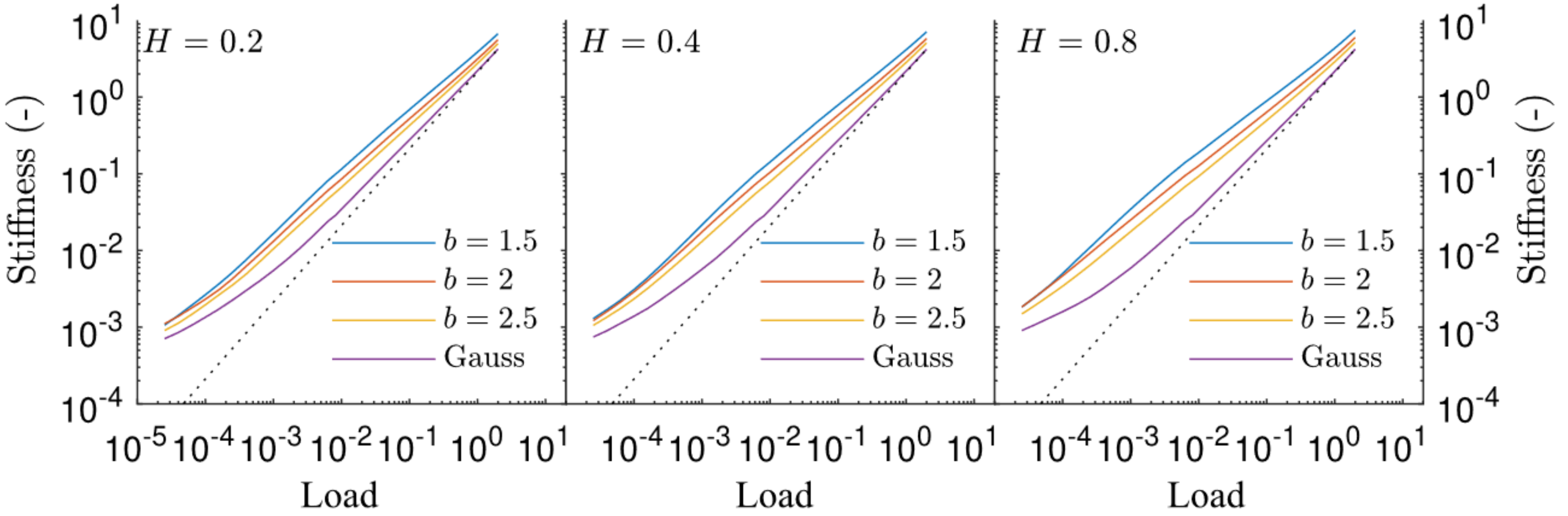


Resulting surface



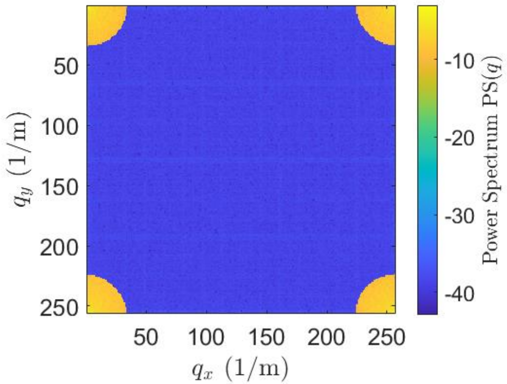
Motivation – Non-Gaussian Surfaces

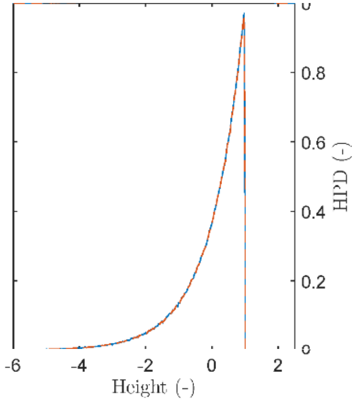
Example of application with Weibull HPD



Pérez-Ràfols, F., Almqvist, A. On the stiffness of surfaces with non-Gaussian height distribution. *Sci Rep* **11**, 1863 (2021).
https://doi.org/10.1038/s41598-021-81259-8

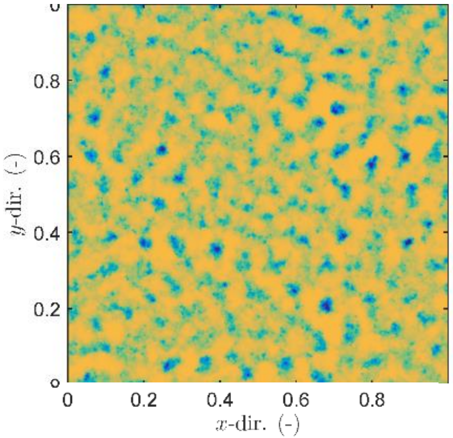
Method – The Surface Randomisation Algorithm





Algorithm

Tolerances



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

Both the PSD and HPD encode $\bar{z}$ and R_q

Normalization guarantee that they are equal

$$Z = \frac{z - \bar{z}}{\mathrm{R_q}}$$

Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

From the Fourier transform of z, we read:

- Modulus: $|\mathcal{F}(z)|$: What we defined as $\sqrt{PSD}$
- Phase: $\angle\mathcal{F}(z)$: Related to HPD

We correct the PS by

$$\mathcal{F}(z)^{n+1} = \mathcal{F}(z)^{n} \frac{\sqrt{PSD}}{|\mathcal{F}(z)^{n}|}$$

This, however, distorted the HPD

Method – The Surface Randomisation Algorithm

<u>Step 1</u>: Normalize the inputs

<u>Step 2</u>: Correct the power spectrum

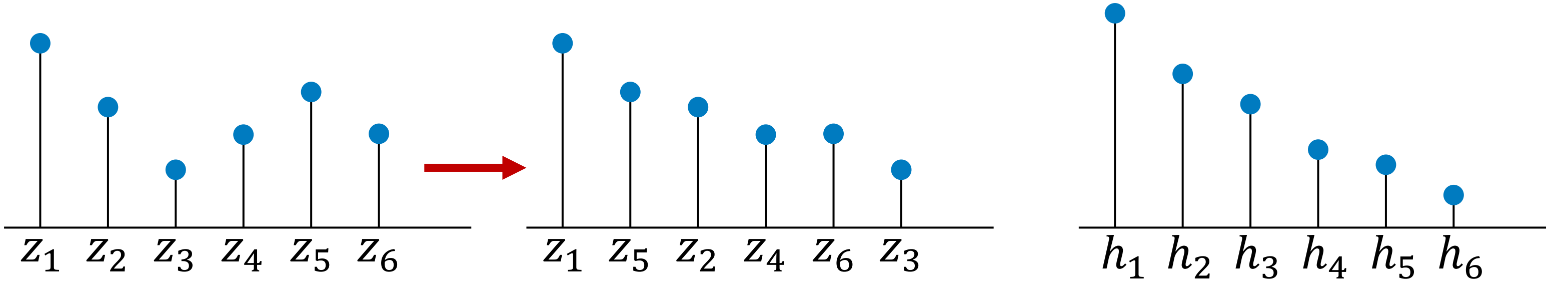
<u>Step 3</u>: Correct the height distribution

<u>Step 4</u>: Compute error

Check exit condition

We want to change the height distribution while minimizing the total change of the surface

- Sort z and h
- Replace, in height order, the value of z by those in h



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

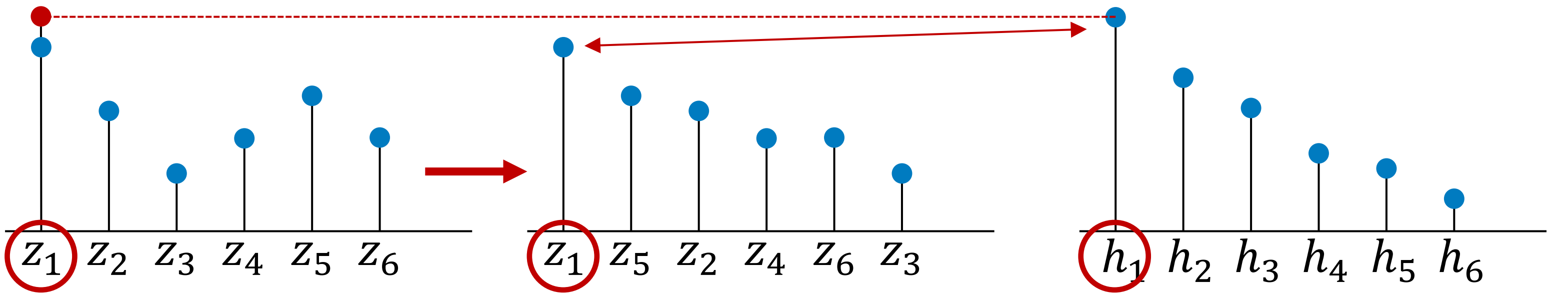
Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

We want to change the height distribution while minimizing the total change of the surface

- Sort z and h
- Replace, in height order, the value of z by those in h



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

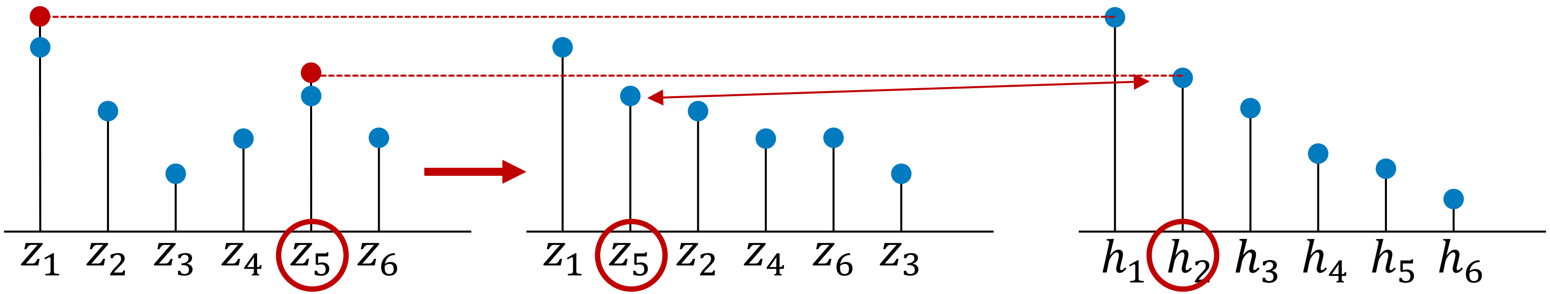
Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

We want to change the height distribution while minimizing the total change of the surface

- Sort z and h
- Replace, in height order, the value of z by those in h



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

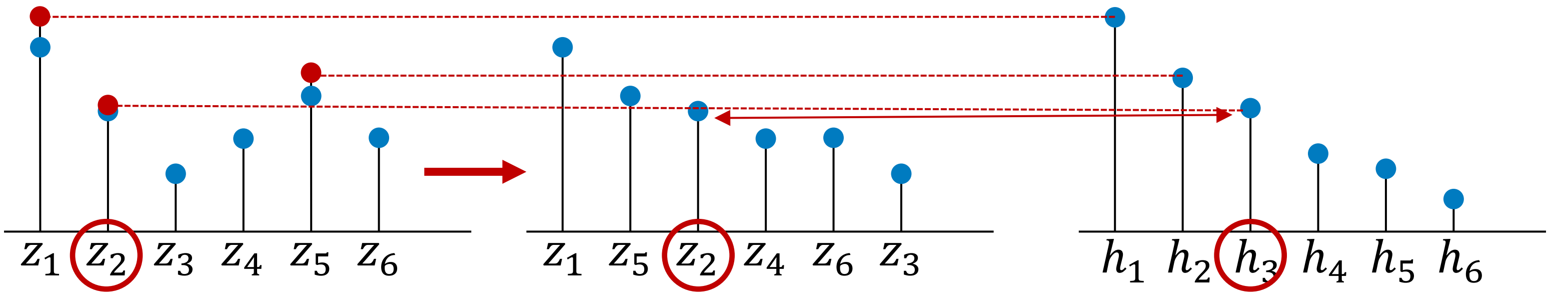
Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

We want to change the height distribution while minimizing the total change of the surface

- Sort z and h
- Replace, in height order, the value of z by those in h



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

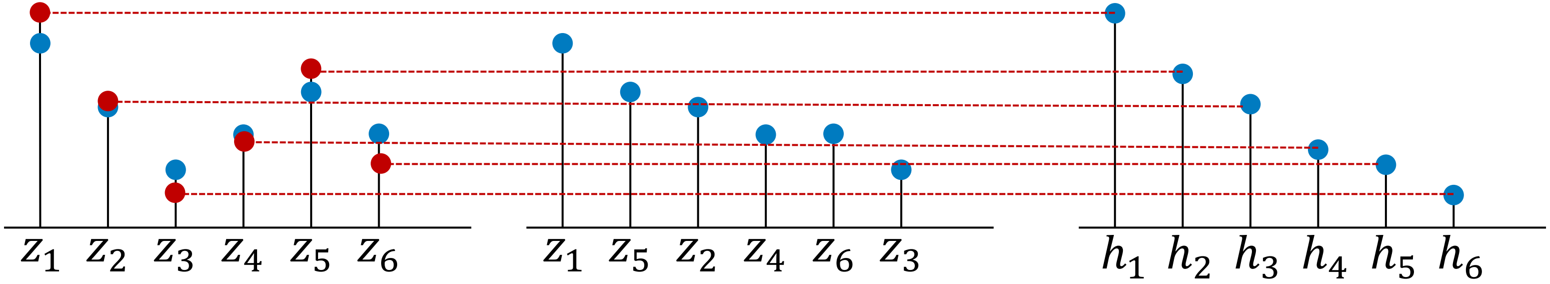
Step 3: Correct the height distribution

Step 4: Compute error

Check exit condition

We want to change the height distribution while minimizing the total change of the surface

- Sort z and h
- Replace, in height order, the value of z by those in h



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height distribution

Step 4: Compute error

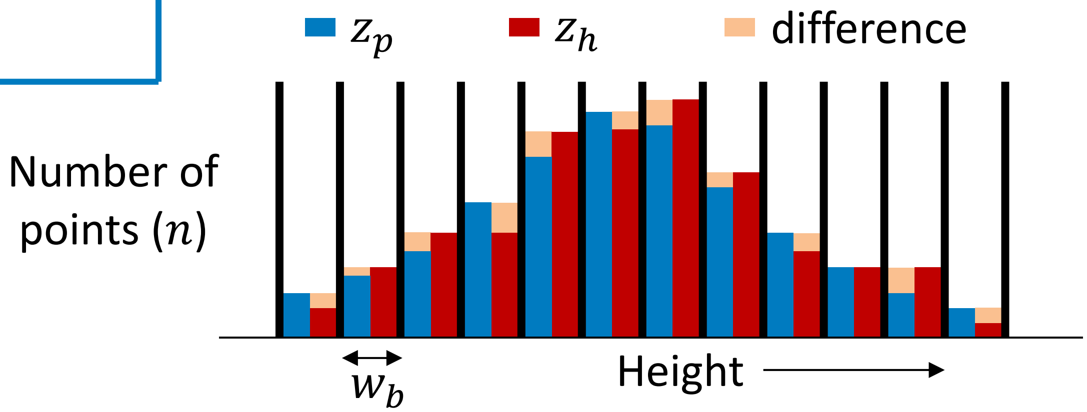
Check exit condition

$$err_h = \sum_{bins} (n_p - n_h) w_b$$

Error in height distribution

z_h has the perfect HPD, we compare it with z_p

- Construct a histogram
- Compare the number of points in each bin



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height

Step 4: Compute error

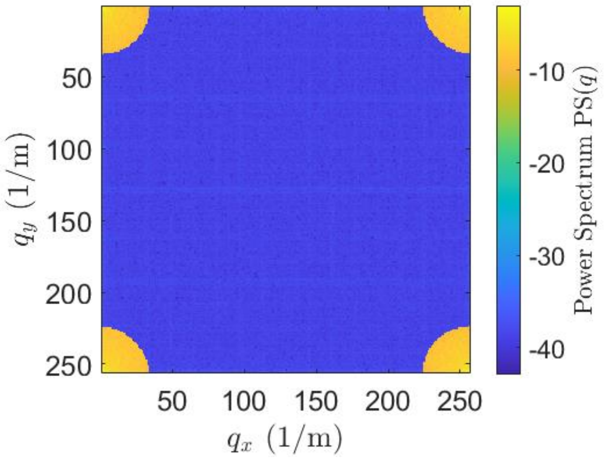
Check exit condition

$$\mathrm{err}_p = \sqrt{\frac{1}{N_S} \sum_{q \mid \mathcal{F}\{z_p\}(q) \neq 0} \left(\frac{|\mathcal{F}\{z_h\}(q)| - |\mathcal{F}\{z_p\}(q)|}{|\mathcal{F}\{z_p\}(q)|} \right)^2}$$

Error in the power spectrum

We use the rms value of the relative error

We must be careful with points that have a prescribed value of zero



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height

Step 4: Compute error

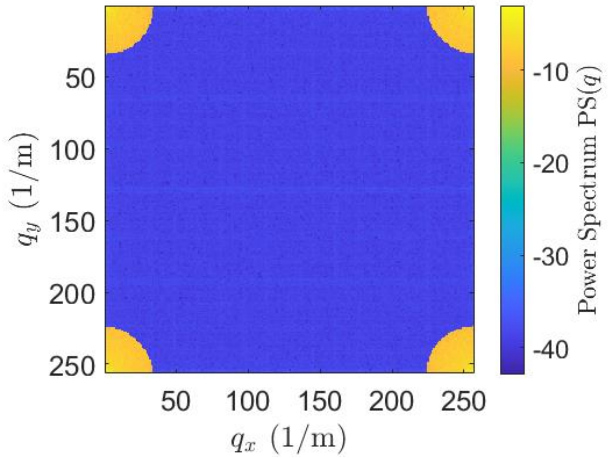
Check exit condition

$$\mathrm{err}_{p_0} = \frac{\sqrt{\frac{1}{N_{S_0}} \sum_{q|\mathcal{F}\{z_p\}(q)=0} |\mathcal{F}\{z_h\}|^2}}{\frac{1}{N_{S_0}} \sum_{q|\mathcal{F}\{z_p\}(q)=0} |\mathcal{F}\{z_p\}|}$$

Error in the power spectrum

We use the rms value of the relative error

We must be careful with points that have a prescribed value of zero



Method – The Surface Randomisation Algorithm

Step 1: Normalize the inputs

Step 2: Correct the power spectrum

Step 3: Correct the height

Step 4: Compute error

Check exit condition

Three exit conditions are used

- Convergence reached

  One of two conditions are fulfilled:
  - $\mathrm{err}_h < \mathrm{tol}$
  - $\mathrm{err}_p < \mathrm{tol}$ and $\mathrm{err}_{p_0} < \mathrm{tol}$

- Slow speed

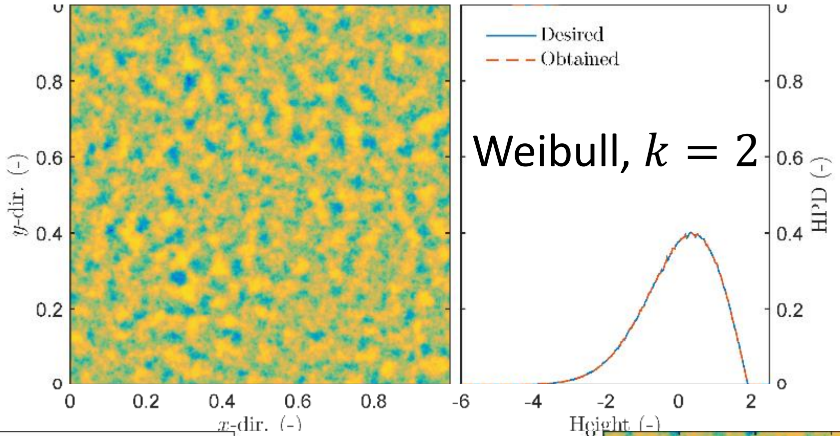
  Convergence is not reached but the surface does not change significantly

- Maximum number of iterations

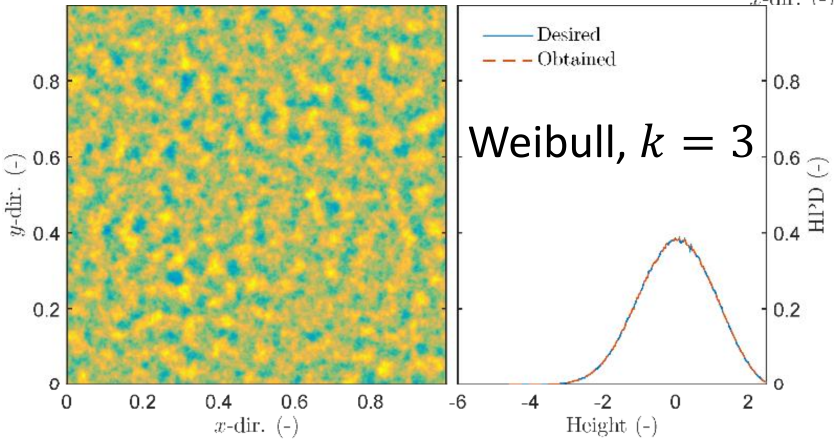
  The algorithm has failed

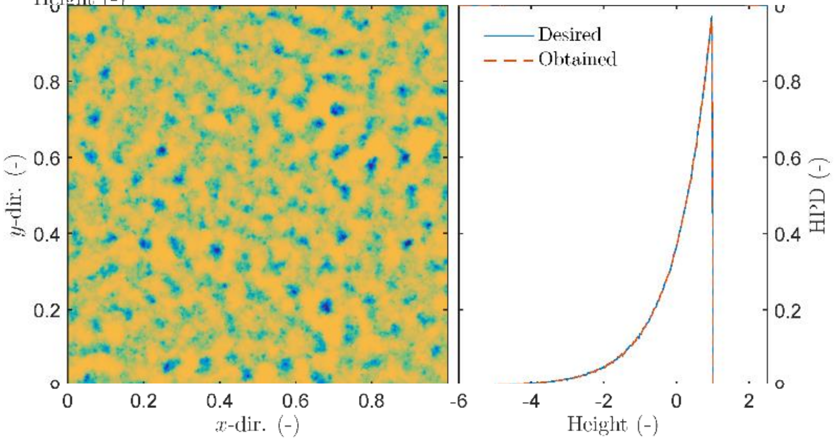
Results – Weibull HPD

Some examples

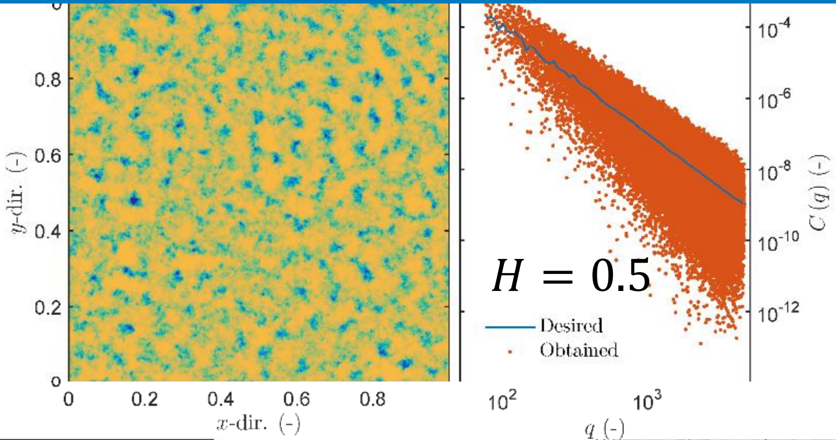


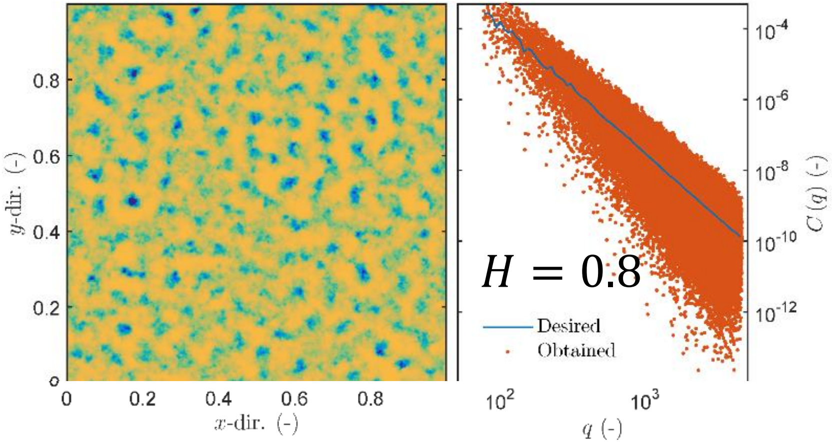
Weibull, $k = 1$

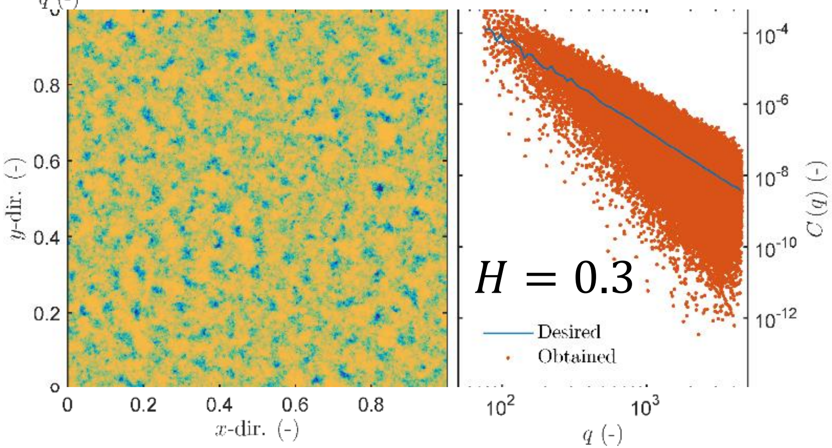




Some examples

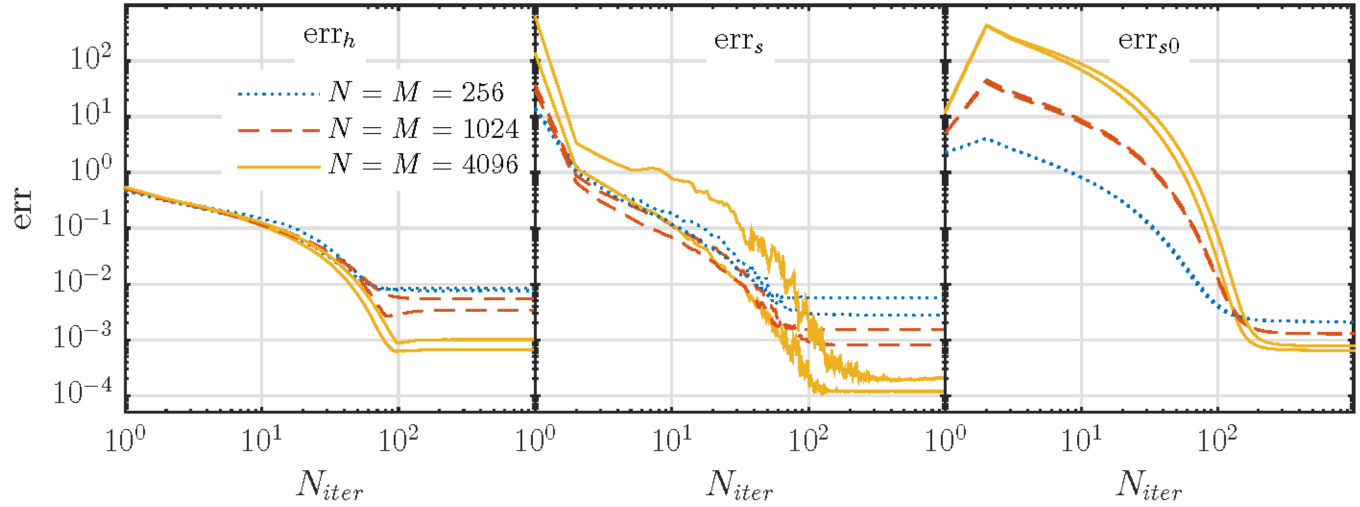




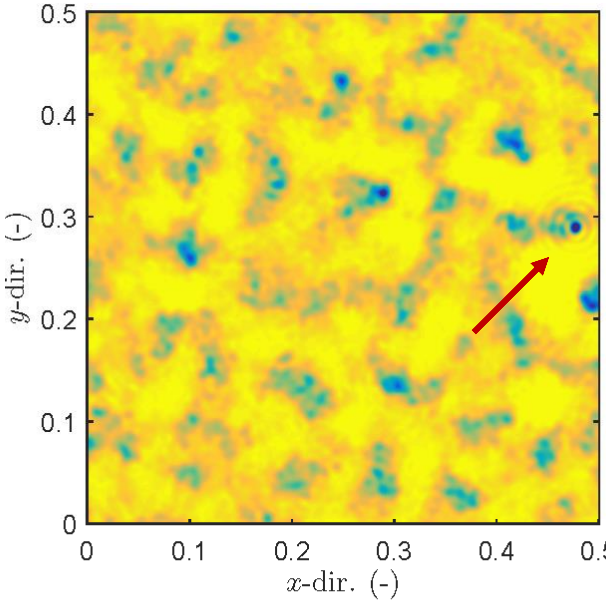


Results – Weibull HPD

Minimum error attainable

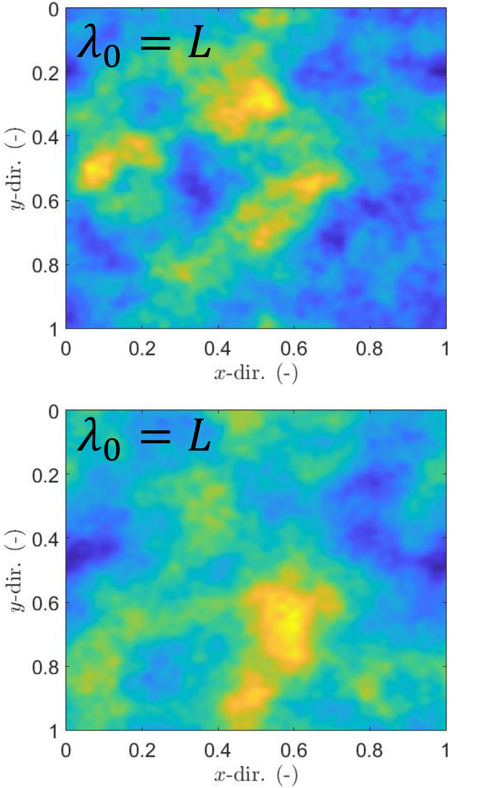


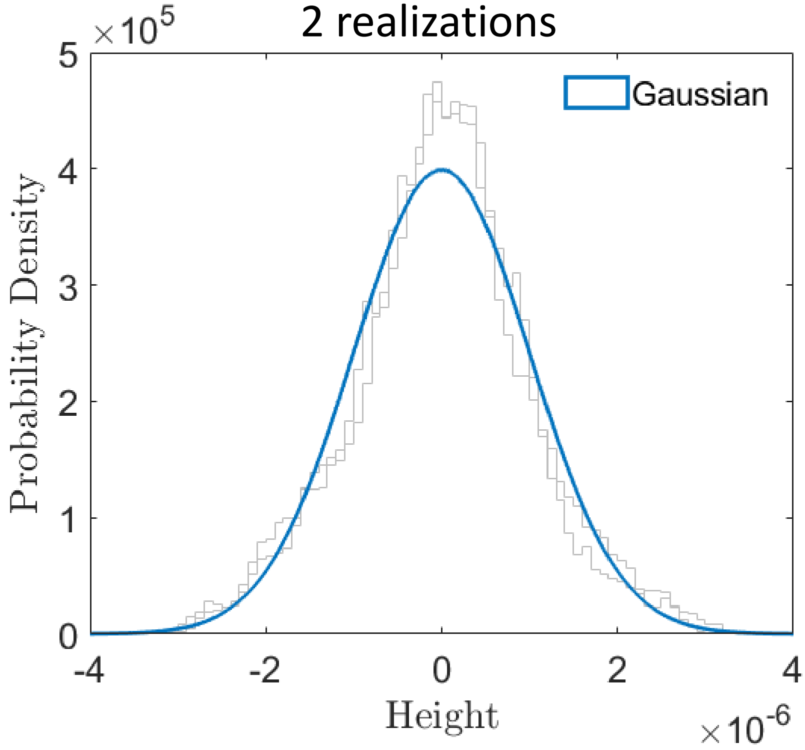
Results – Artefacts



Results – Gaussian Example

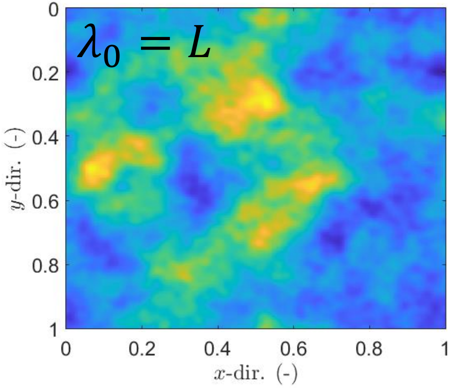
Giving sufficient flexibility for HPD and PSD to match

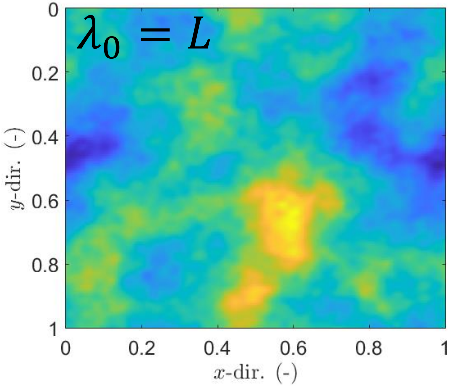


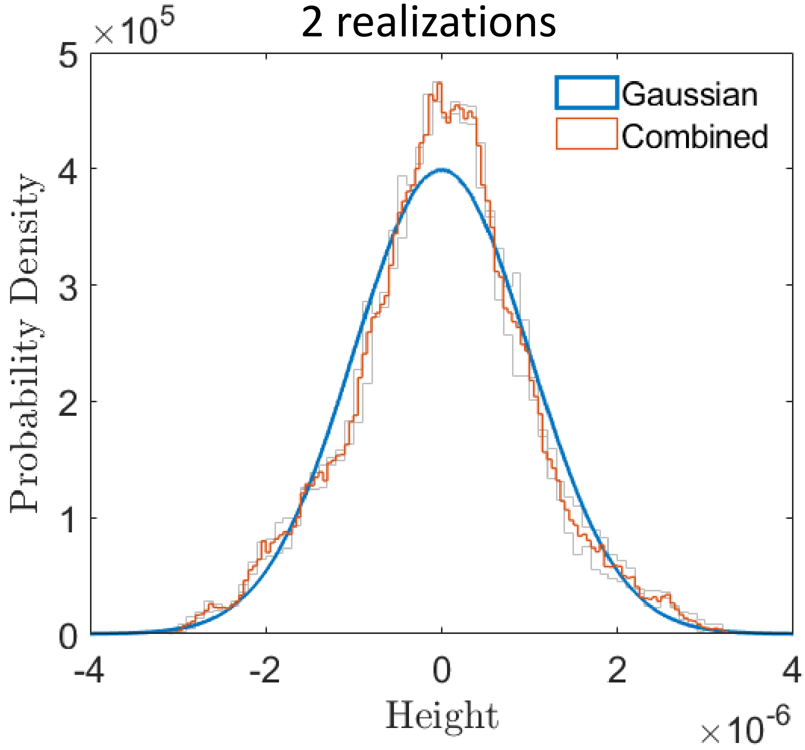


Results – Gaussian Example

Giving sufficient flexibility for HPD and PSD to match

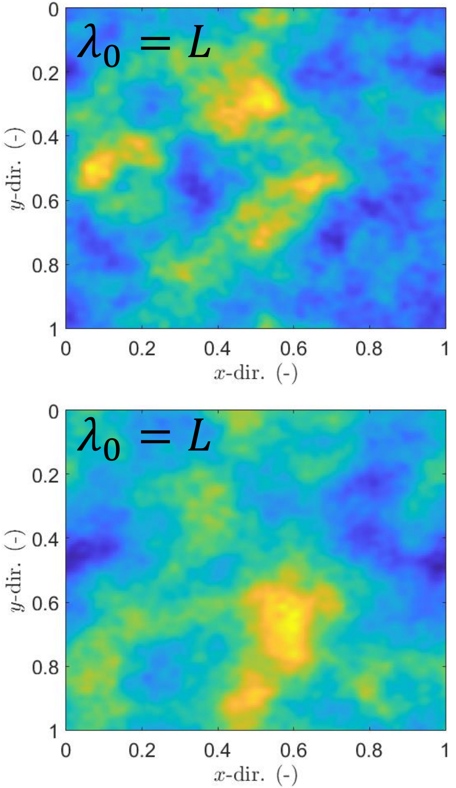


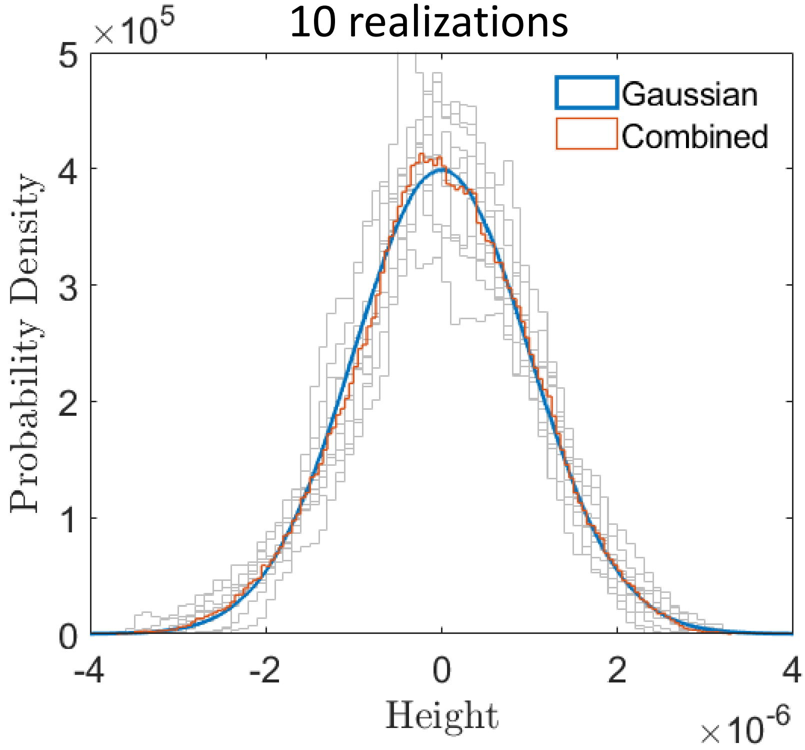




Results – Gaussian Example

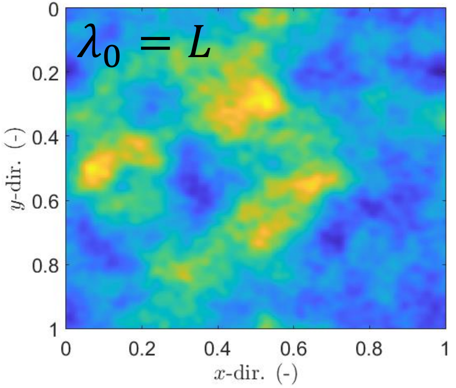
Giving sufficient flexibility for HPD and PSD to match

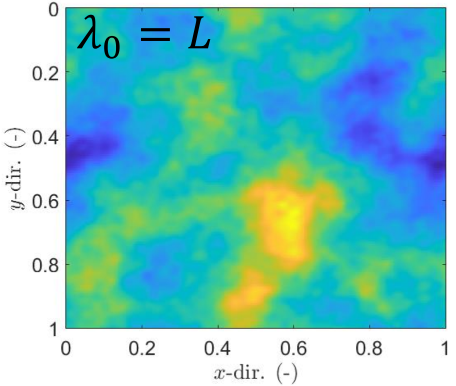


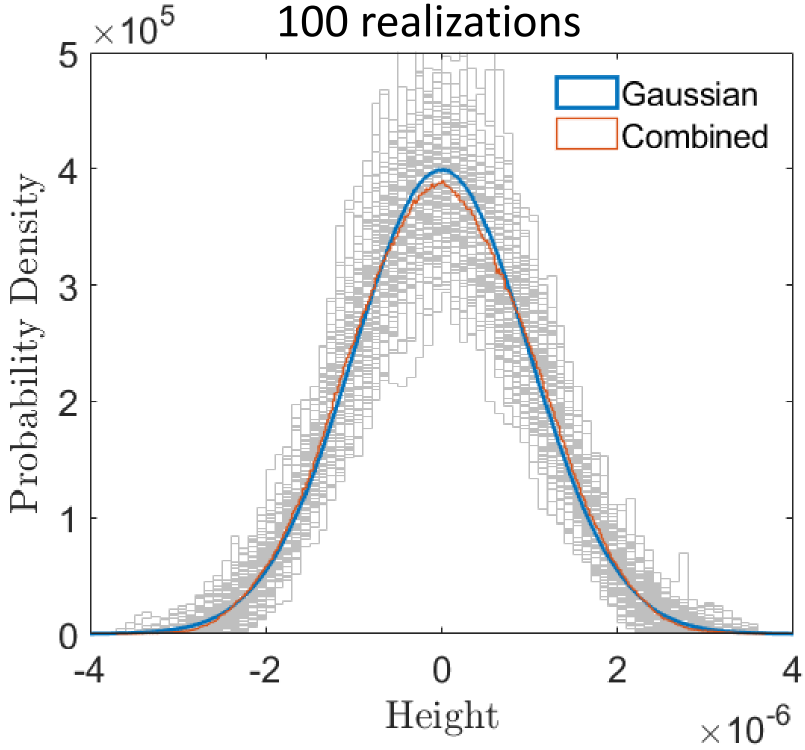


Results – Gaussian Example

Giving sufficient flexibility for HPD and PSD to match

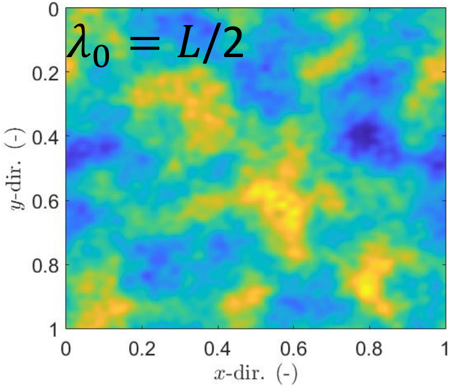


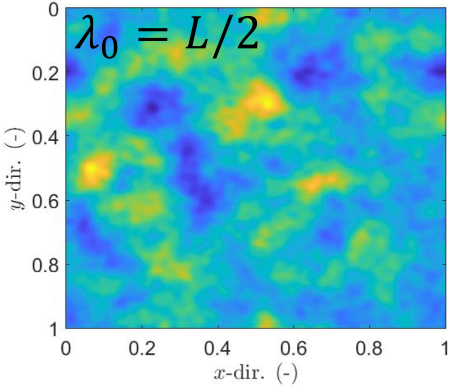


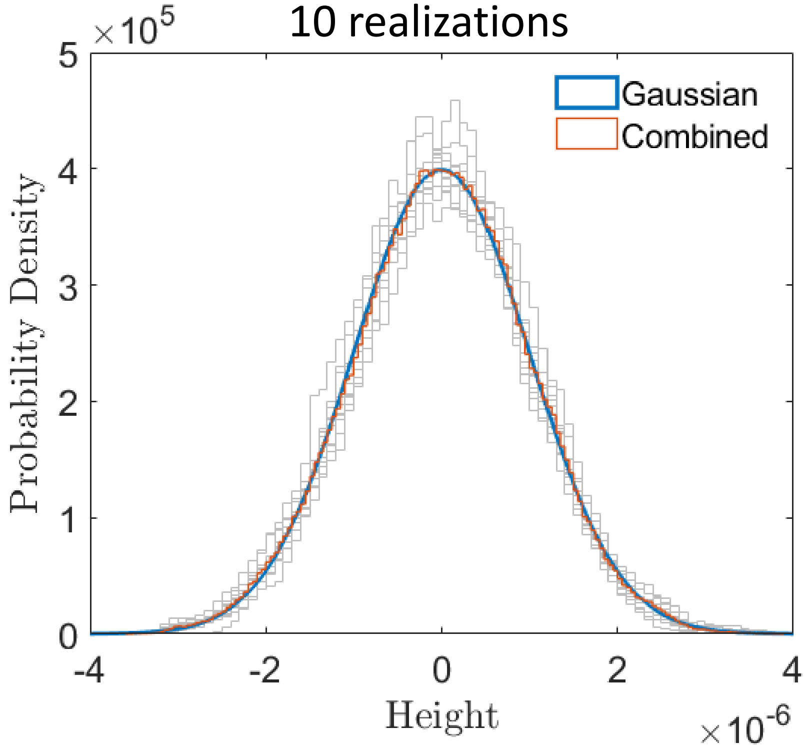


Results – Gaussian Example

Giving sufficient flexibility for HPD and PSD to match

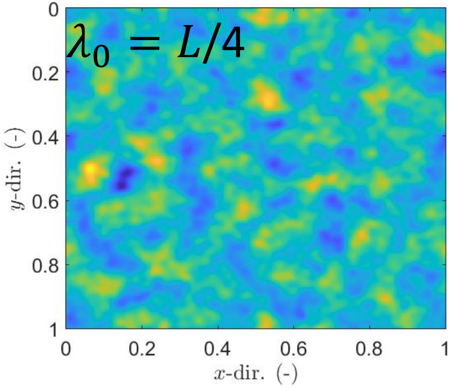


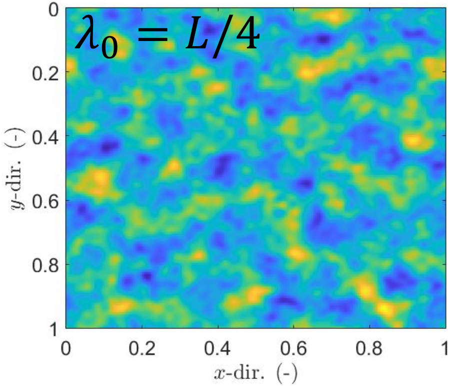


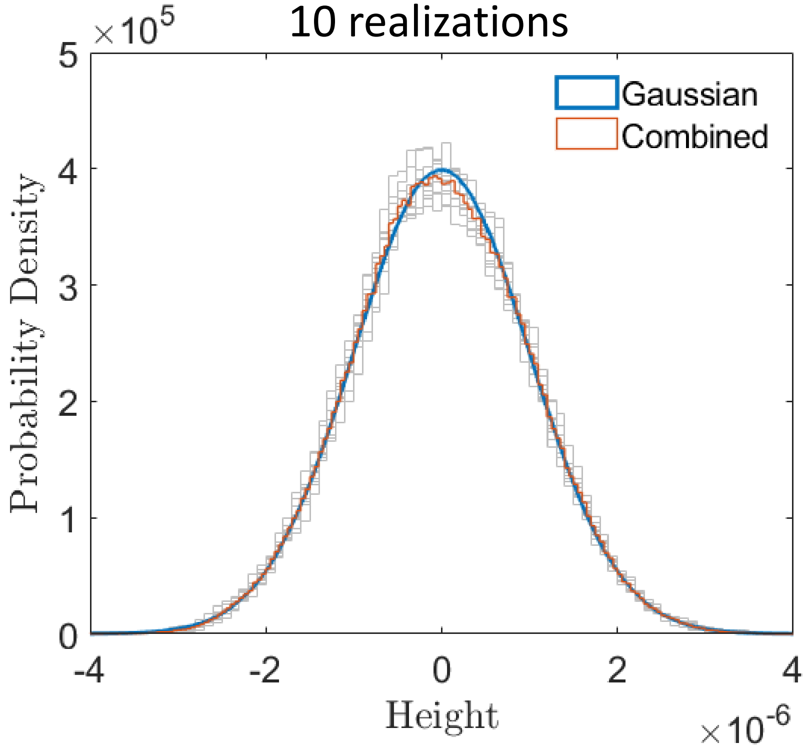


Results – Gaussian Example

Giving sufficient flexibility for HPD and PSD to match

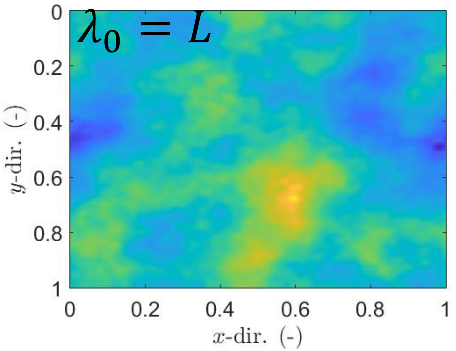






Results – Gaussian Example

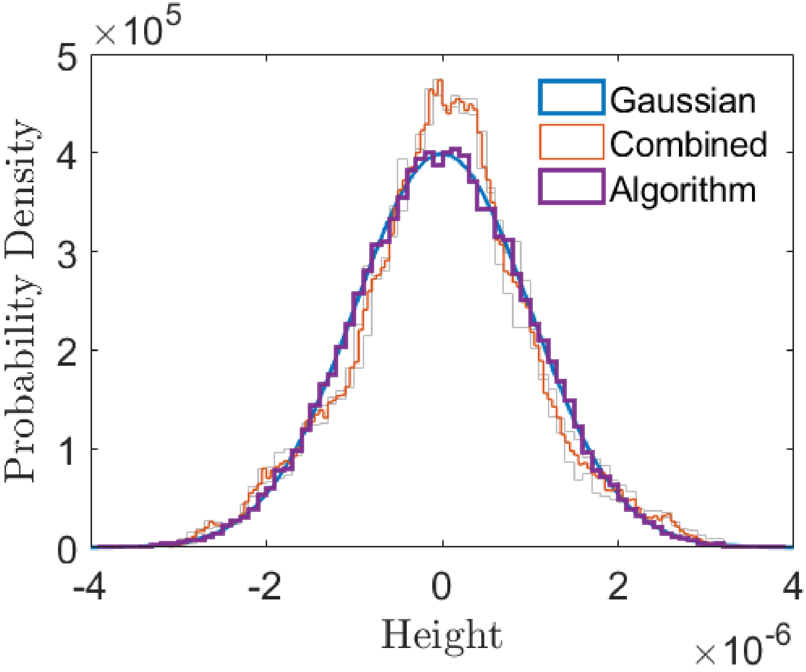
Giving sufficient flexibility for HPD and PSD to match



The code is too powerful!

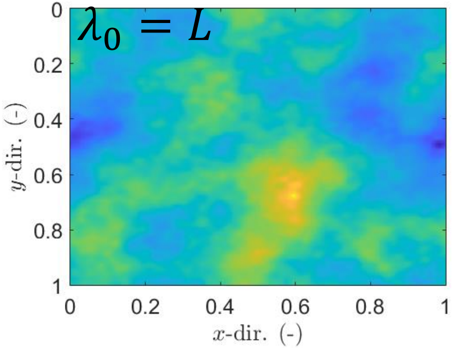
If you ask it to give a perfect HPD with $\lambda_0 = L$, it will often provide, but artefacts might appear

2 realizations



Results – Gaussian Example

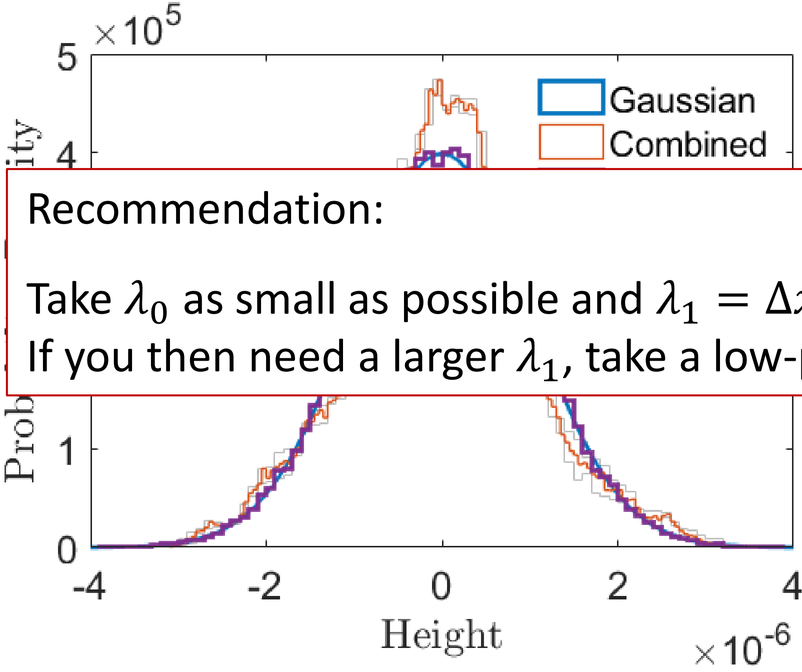
Giving sufficient flexibility for HPD and PSD to match



The code is too powerful!

If you ask it to give a perfect HPD with $\lambda_0 = L$, it will often provide, but artefacts might appear

2 realizations



Recommendation:

Take λ_0 as small as possible and $\lambda_1 = \Delta x$

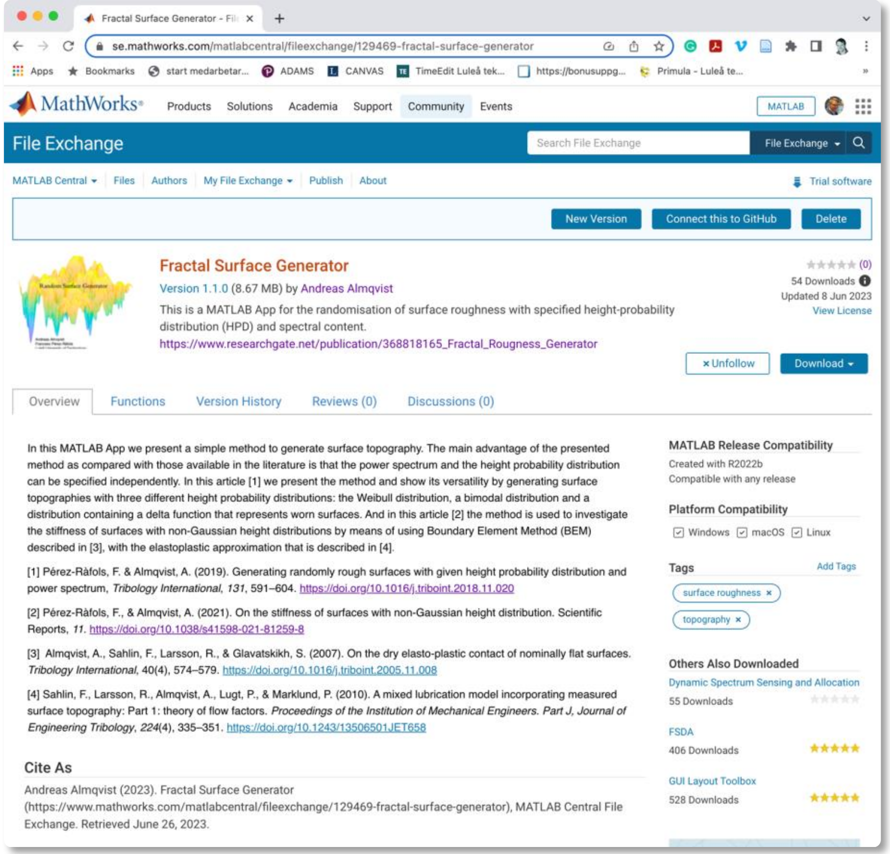
If you then need a larger λ_1, take a low-pass filter

Summary

- Generated surfaces can be a powerful tool to study the effect of roughness
- These are even more useful if they can be flexibly generated, controlling both power spectrum and height probability distribution
- We have presented an algorithm that achieves this flexibility and is easy to operate
  - Caveat: one must be careful with cut-off wavelengths

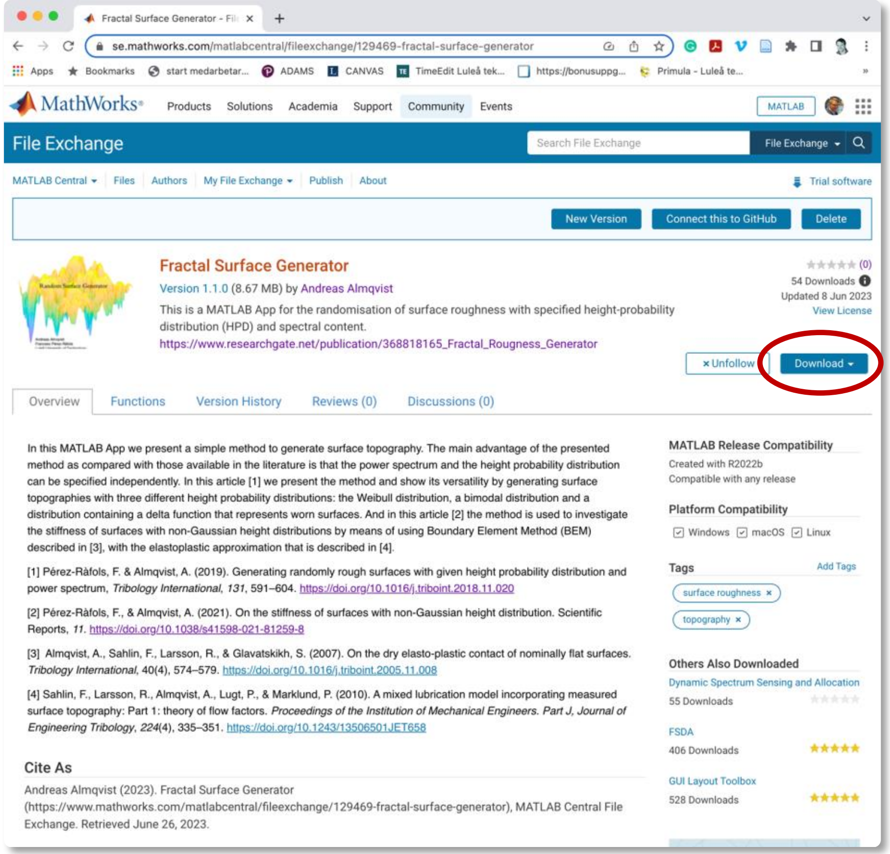
The MATLAB App

1. Go to:
   https://bit.ly/surfgenFPRAA



The MATLAB App

1. Go to: https://bit.ly/surfgenFPRAA
2. Use the Download ▾ button

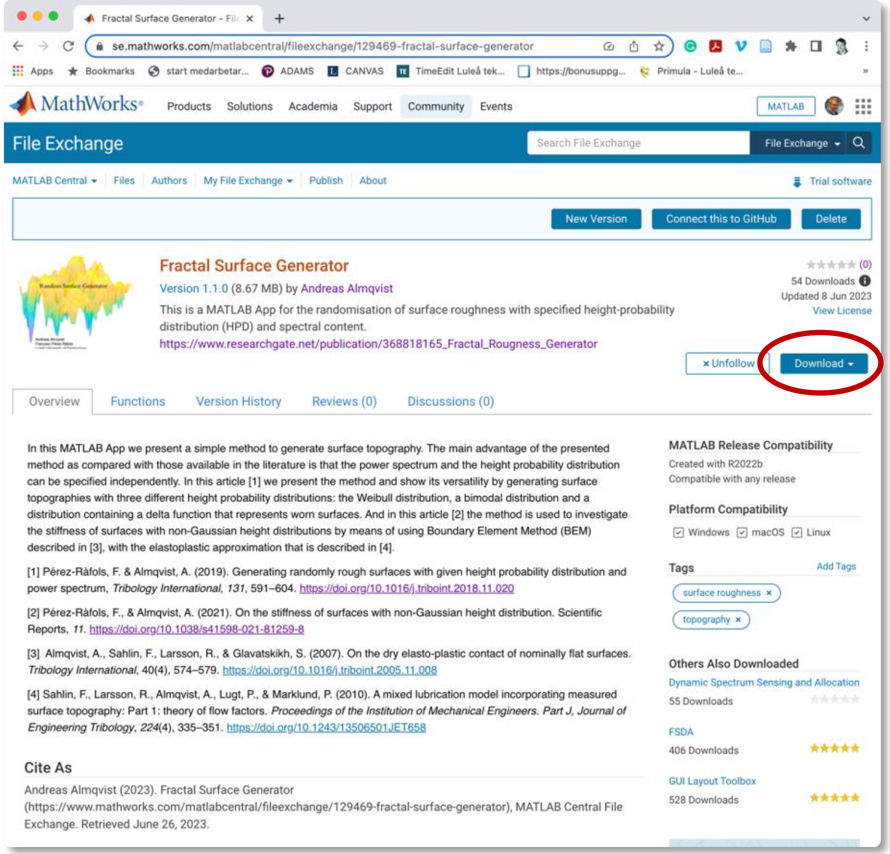


The MATLAB App

1. Go to: https://bit.ly/surfgenFPRAA
2. Use the Download ▾ button

   Toolbox
   Zip

   and the Zip option



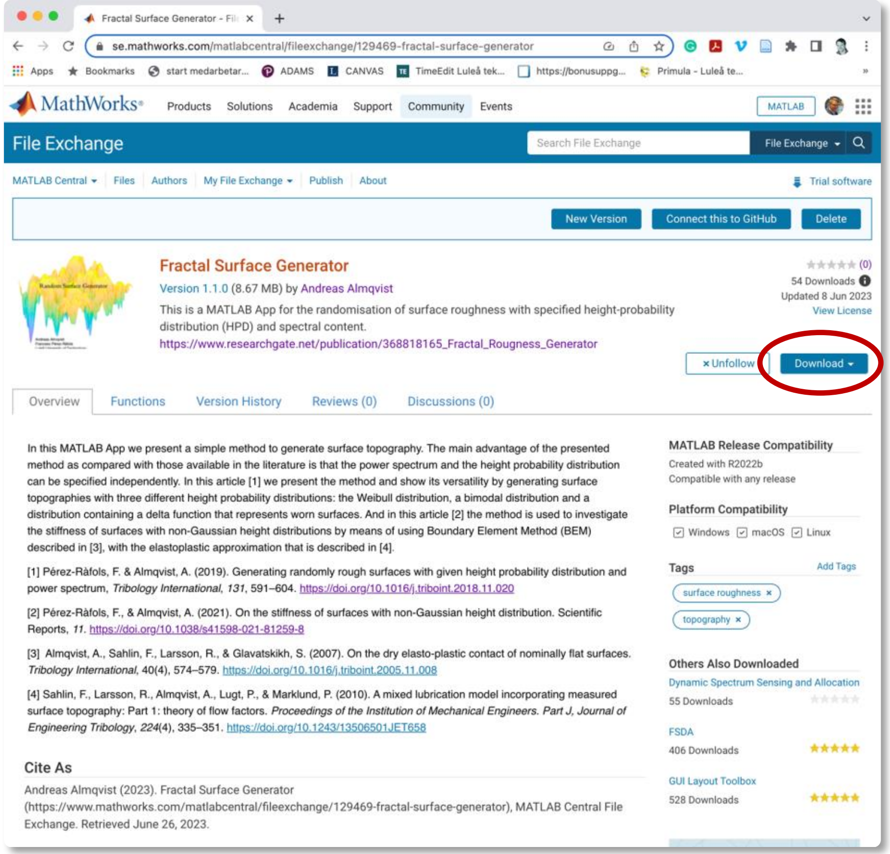
The MATLAB App

1. Go to: https://bit.ly/surfgenFPRAA
2. Use the Download ▾ button

   Toolbox

   Zip

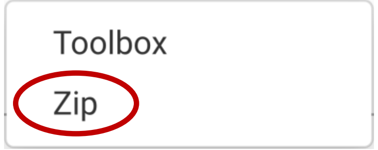
   and the Zip option
3. Unzip (to preferred directory)



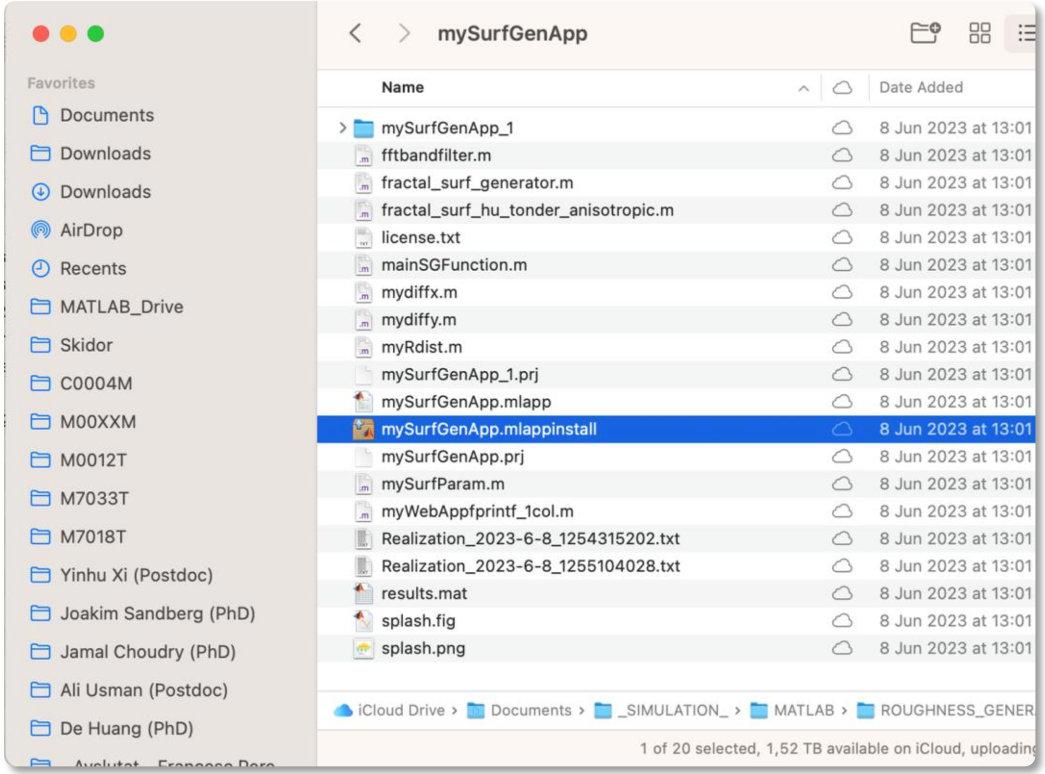
The MATLAB App

1. Go to: https://bit.ly/surfgenFPRAA
2. Use the 

button

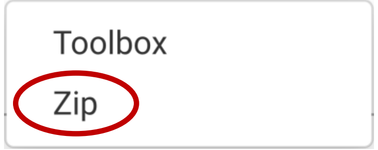


and the Zip option
3. Unzip (to preferred directory)
4. Start MATLAB
5. Run mySurfGenApp.mlappinstall from the directory chosen

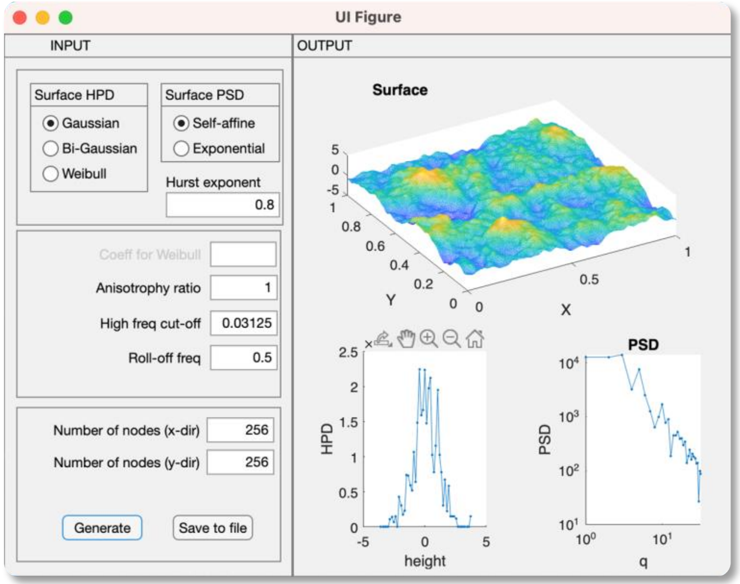


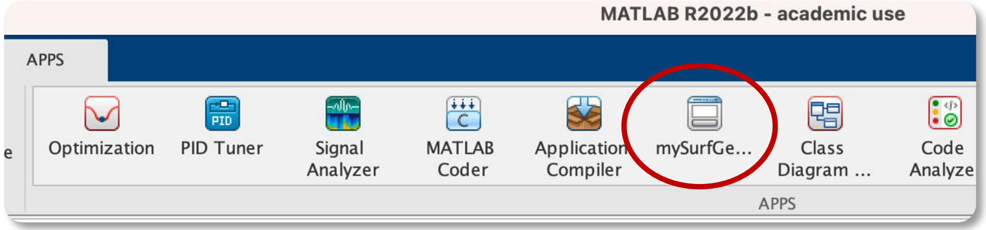
The MATLAB App

1. Go to: https://bit.ly/surfgenFPRAA
2. Use the Download button

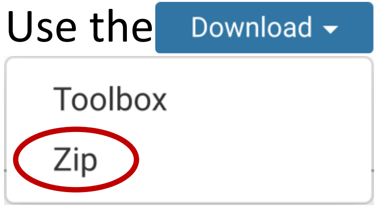


   and the Zip option
3. Unzip (to preferred directory)
4. Start MATLAB
5. Run mySurfGenApp.mlappinstall from the directory chosen
6. Locate mySurfGe...on the APPS tab and click it!

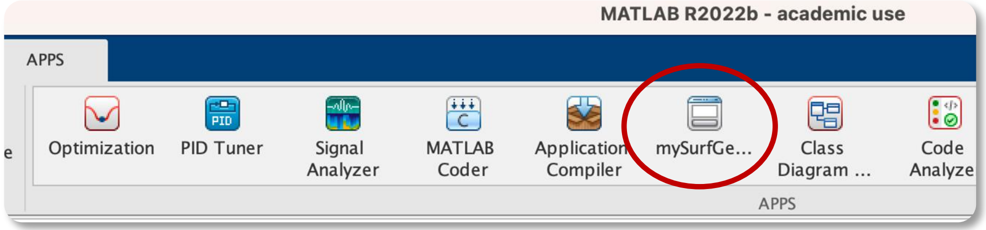




The MATLAB App

1. Go to: https://bit.ly/surfgenFPRAA
2. Use the 

button and the Zip option
3. Unzip (to preferred directory)
4. Start MATLAB
5. Run mySurfGenApp.mlappinstall from the directory chosen
6. Locate mySurfGe…on the APPS tab



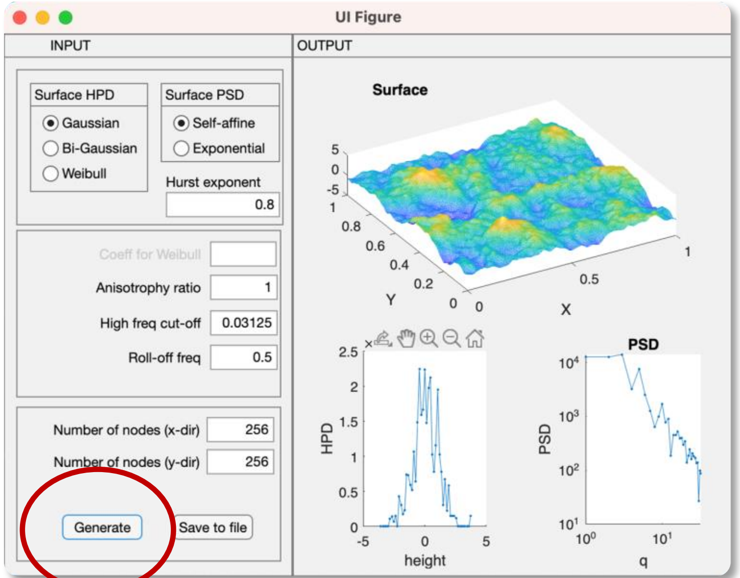
The MATLAB App

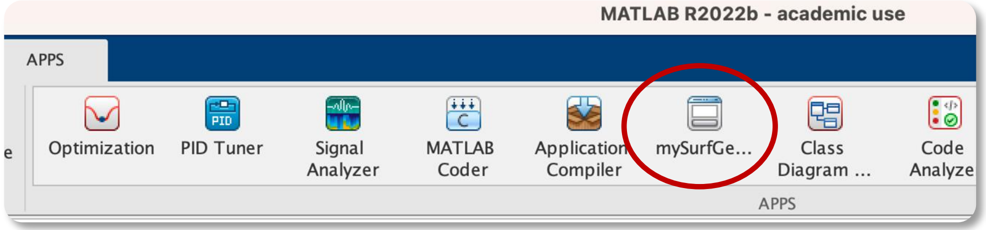
1. Go to: https://bit.ly/surfgenFPRAA
2. Use the Download ▾ button

   Toolbox

   Zip

   and the Zip option
3. Unzip (to preferred directory)
4. Start MATLAB
5. Run mySurfGenApp.mlappinstall from the directory chosen
6. Locate mySurfGe…on the APPS tab
7. Click it and then Generate





The MATLAB App

- Now play with the App as you prefer!
- Notice that you also can use the functions:
  mainSGFunction.m
  mySurfParam.m
  mySurfGenApp.mlappinstall
  standalone