赤方偏移空間における
ハロークラスタリングのエミュレータ構築

東京大学 理学系研究科物理学専攻 M2

小林洋祐

collaborated with
西道啓博 & 高田昌広

in Kavli IPMU

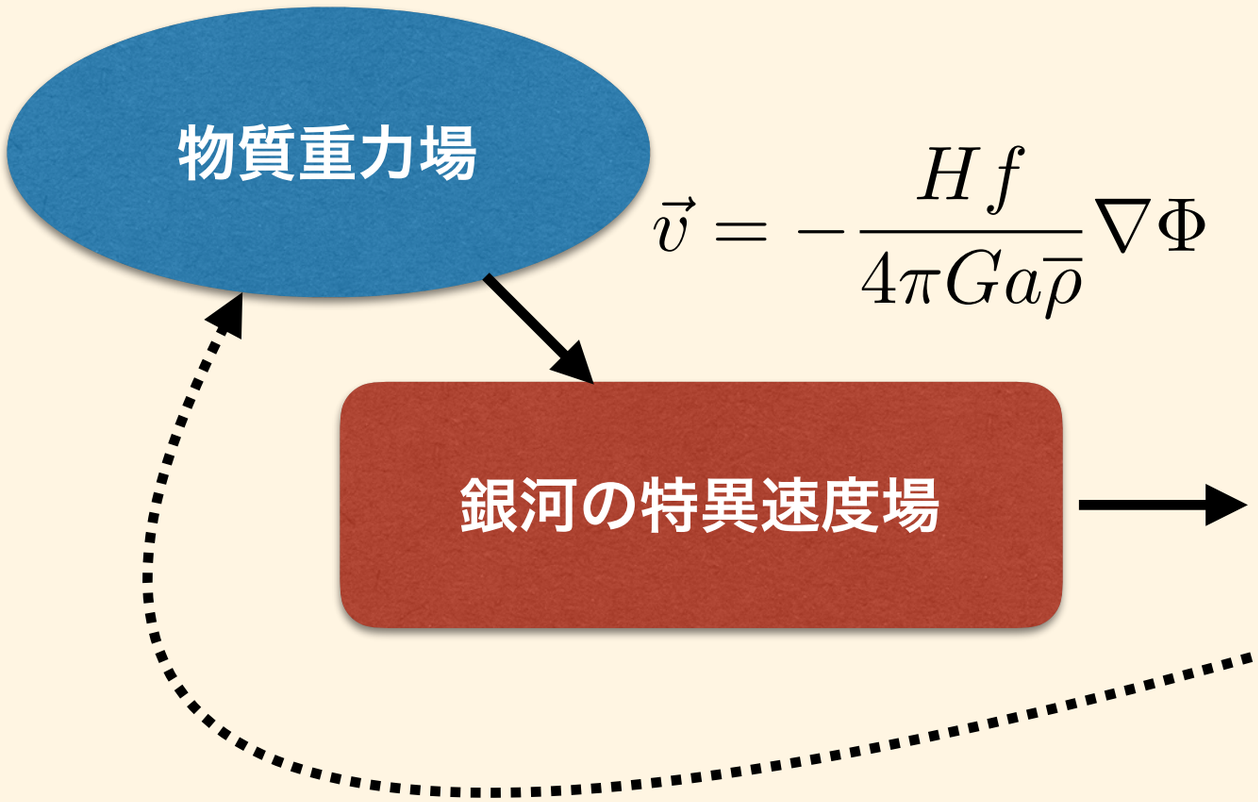
第6回観測的宇宙論ワークショップ@弘前大学

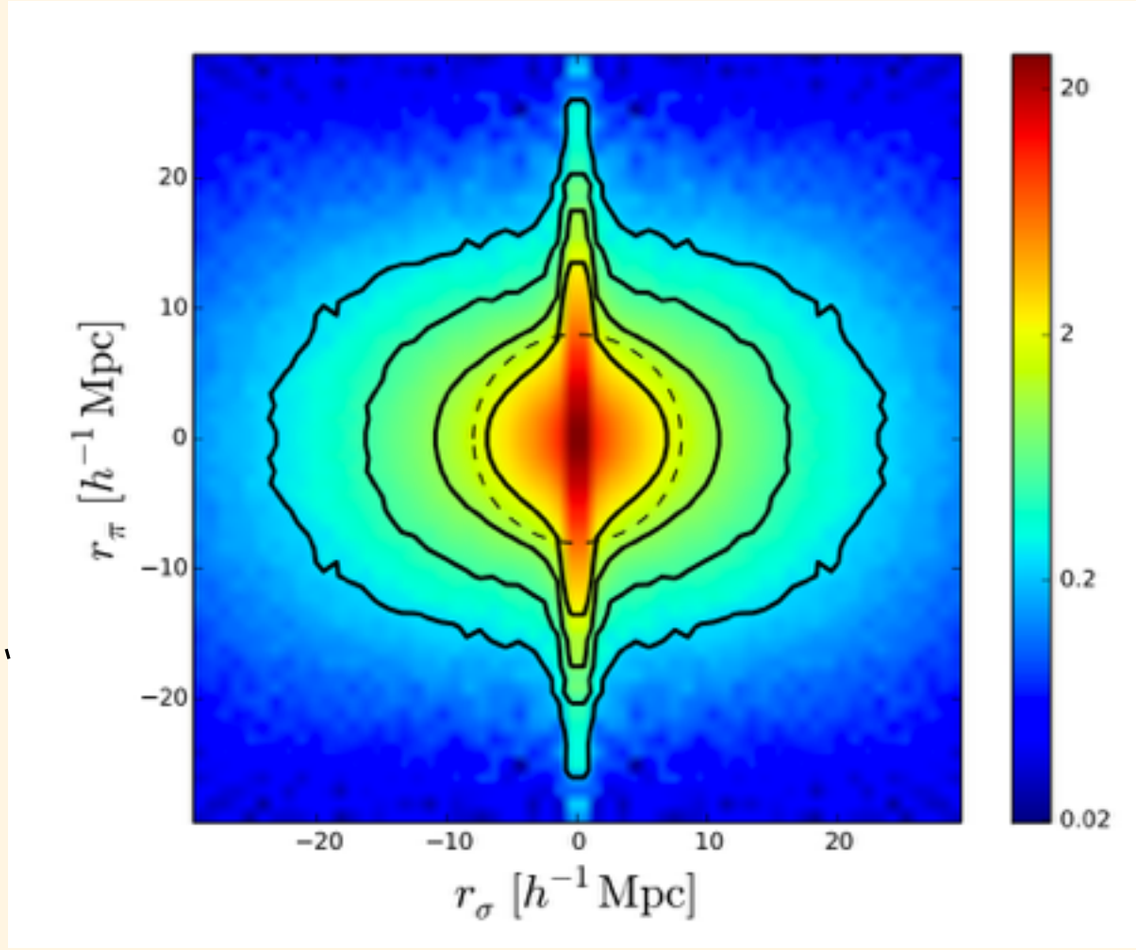
Clustering in Redshift Space

- 銀河分光サーベイで測定される銀河の赤方偏移＝

  Hubble膨張 ＋ 特異速度場の視線方向成分

- 赤方偏移空間・・・視線方向に歪んだ非等方クラスタリング＝赤方偏移歪み





赤方偏移空間のクラスタリングの解析から
重力の情報が得られる

Reid et al. (2014)

Modeling of Clustering in Redshift Space

- 赤方偏移空間クラスタリングの理論モデリングの問題

  'Finger-of-God' effect ... 小スケールでのビリアル速度

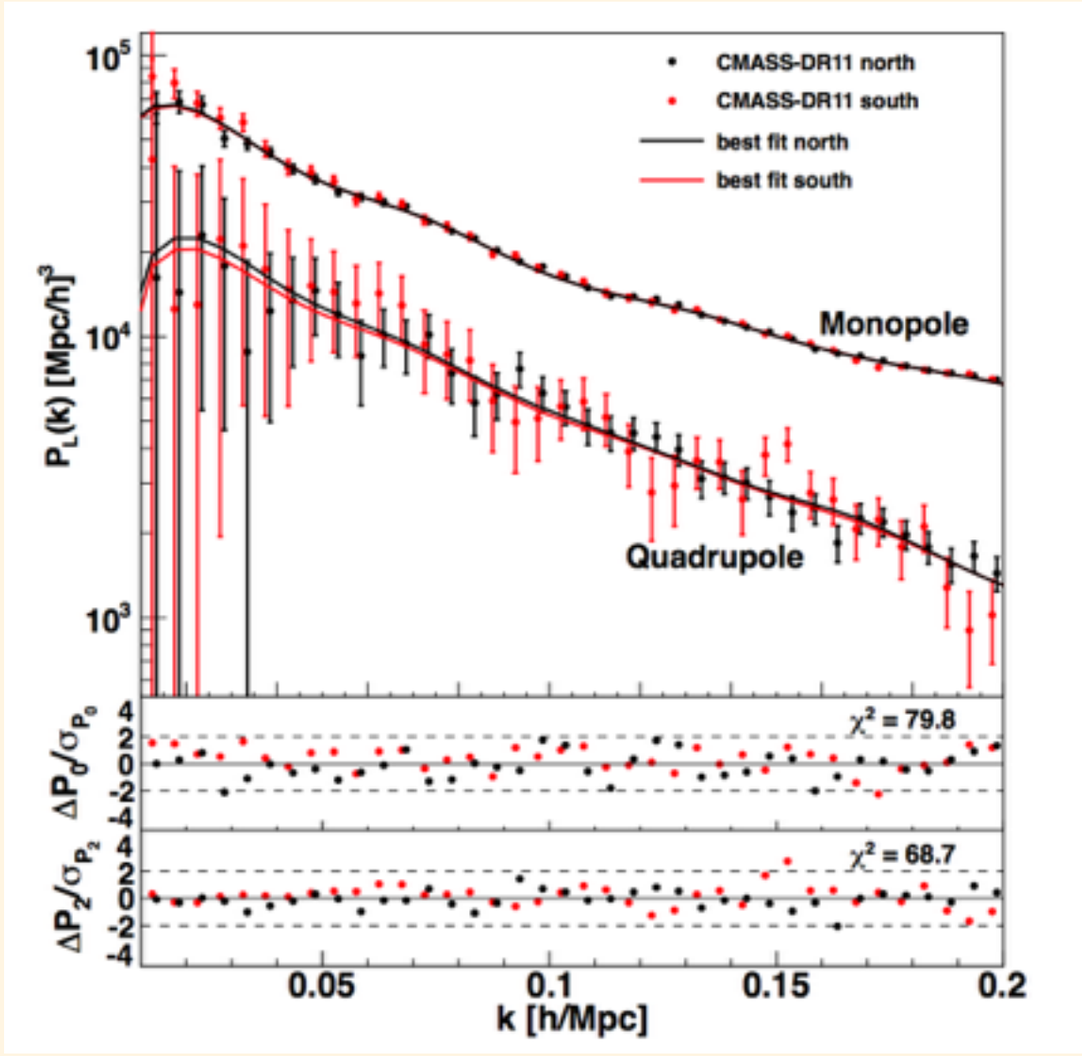
  Galaxy bias ?? ... 銀河形成過程の不定性

- 非線形スケール（k > 0.2 h/Mpc）で
  精密な解析的模型の構築が困難

- N体シミュレーションに基づき、ハローの
  クラスタリングをモデリング

$$P_l(k) = (2l+1)\int_{-1}^{1} P_s(k,\mu)\mathcal{L}_l(\mu)\frac{d\mu}{2}$$

視線方向との角度依存性をLegendre展開、
0次と２次のモーメントが主要観測量

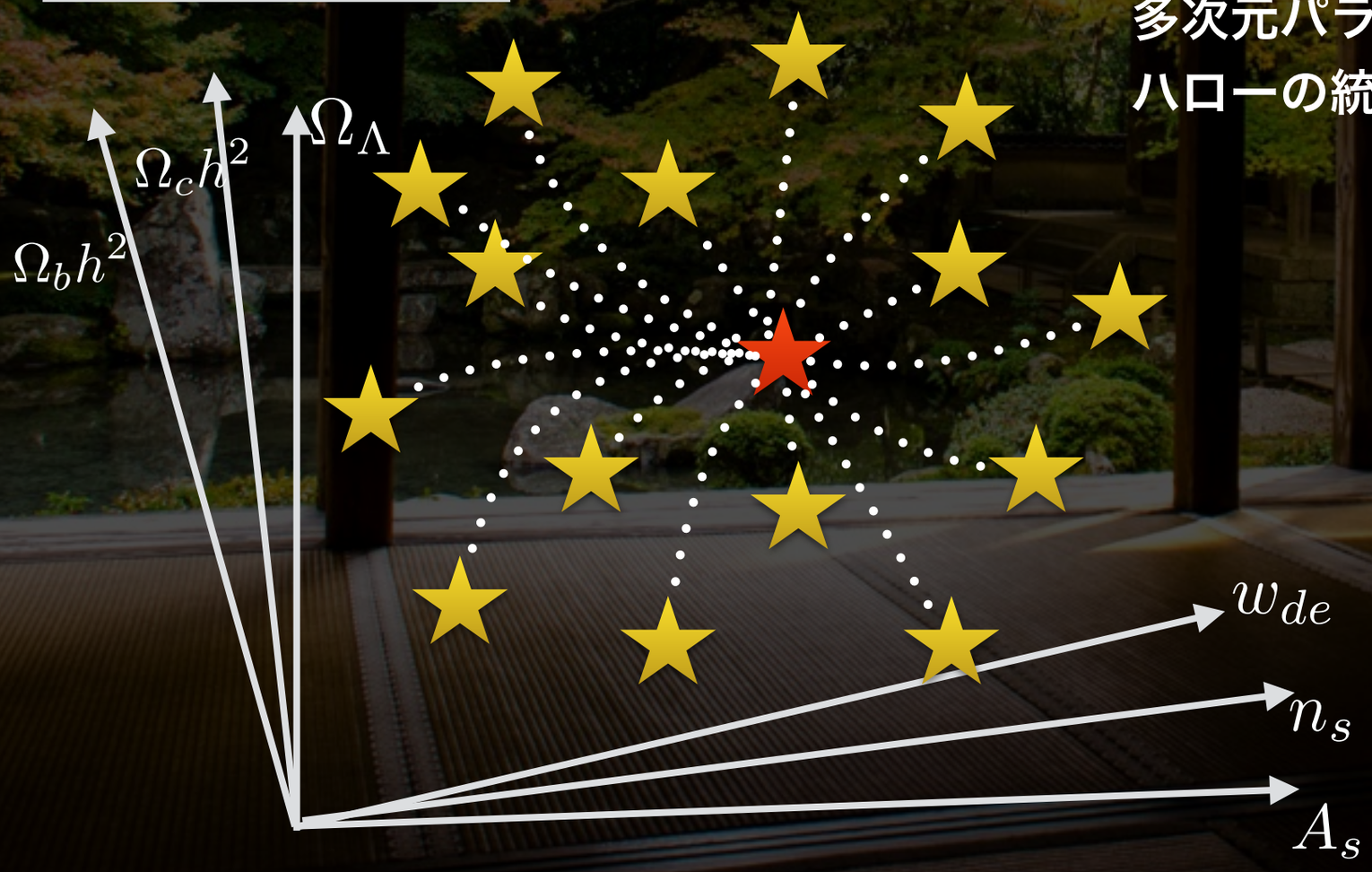


Beutler et al. (2014)

Dark Emulator Project

- 東大 Kavli IPMU の西道さん主導
- 構造形成N体シミュレーション＋機械学習によるハローの宇宙論統計量の理論模型構築

parameter space



多次元パラメータ空間上の離散点で計算した
ハローの統計量を学習、他の点での値を予言

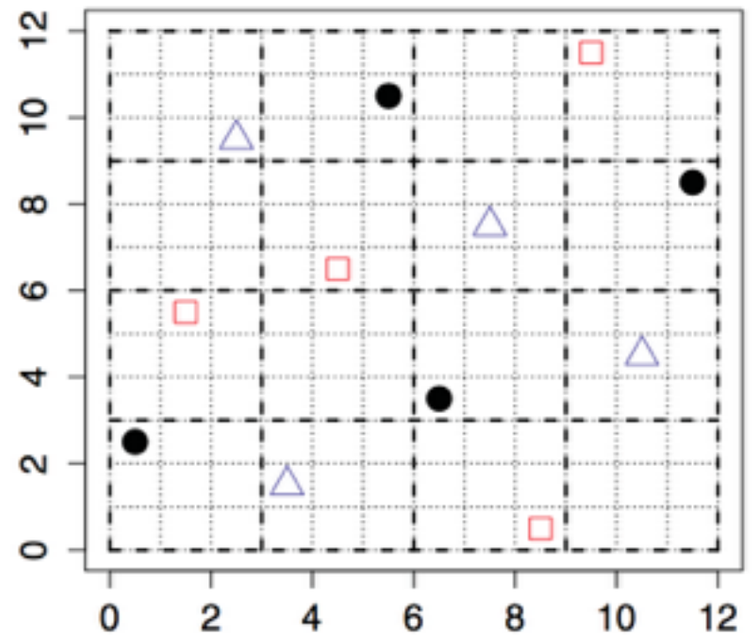
6D 宇宙論パラメータ、
赤方偏移、ハロー最小質量
に対する依存性を
シミュレーションデータから
読み取る

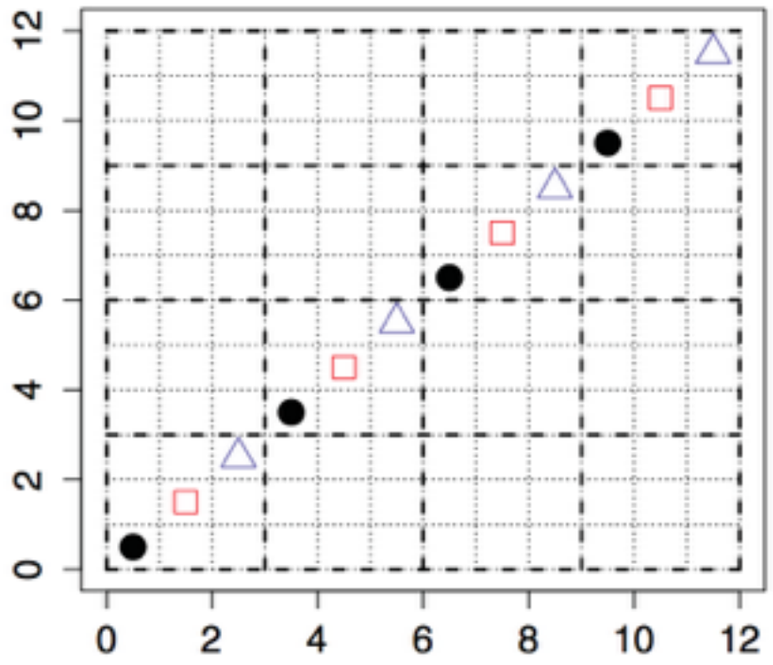
Dark Emulator Project

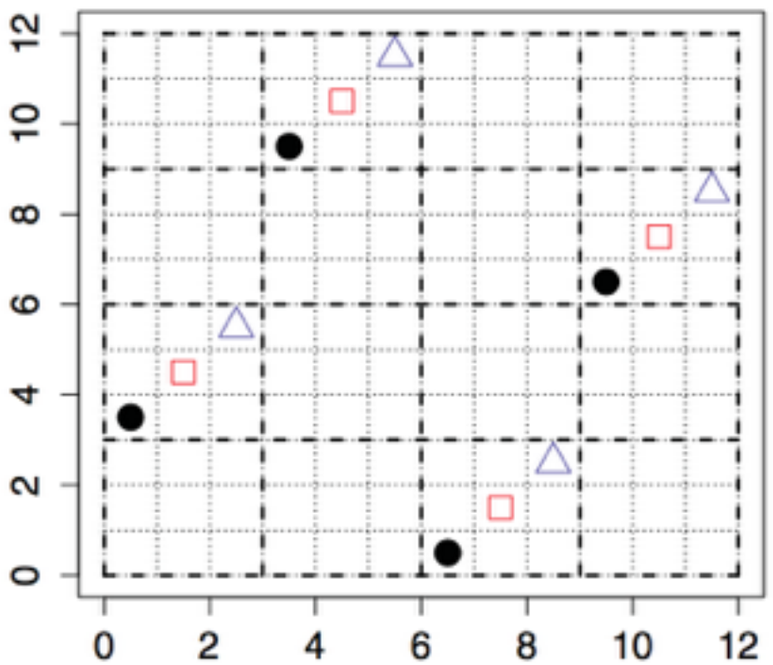
Optimal Sliced Latin Hypercube Design

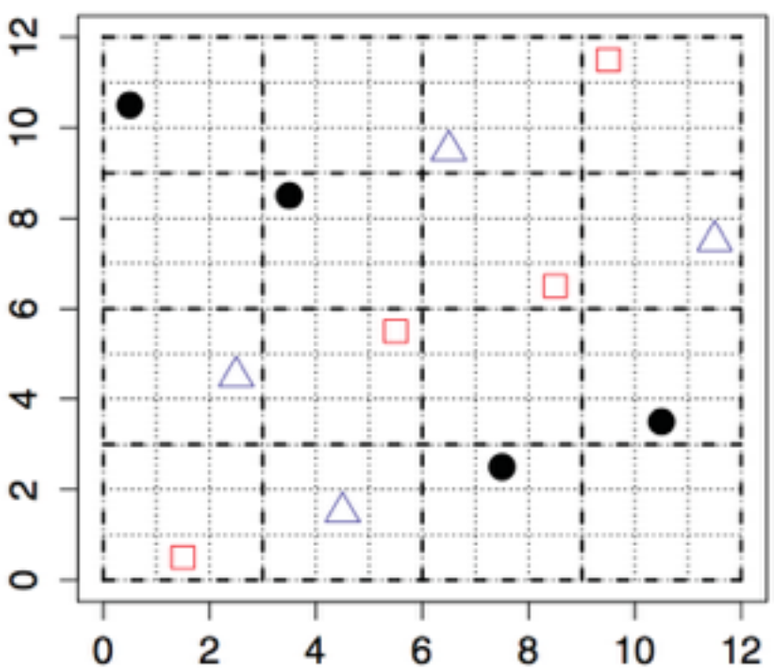
（S. Ba et al, 2015）

- パラメータ空間から偏りなく効率的にサンプリング
- 最近傍点との距離の総和を最大化するように各パラメータの値を組み合わせる







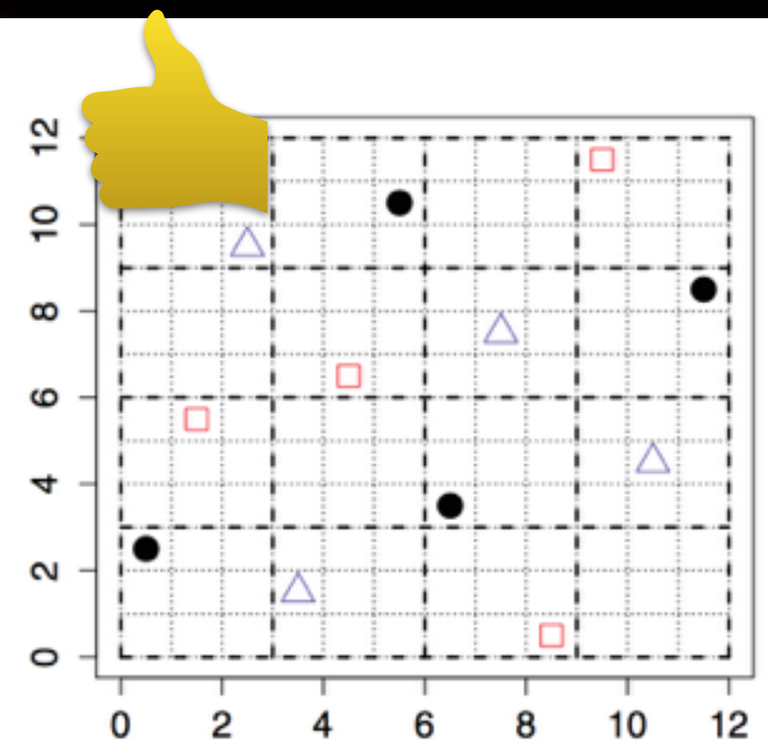


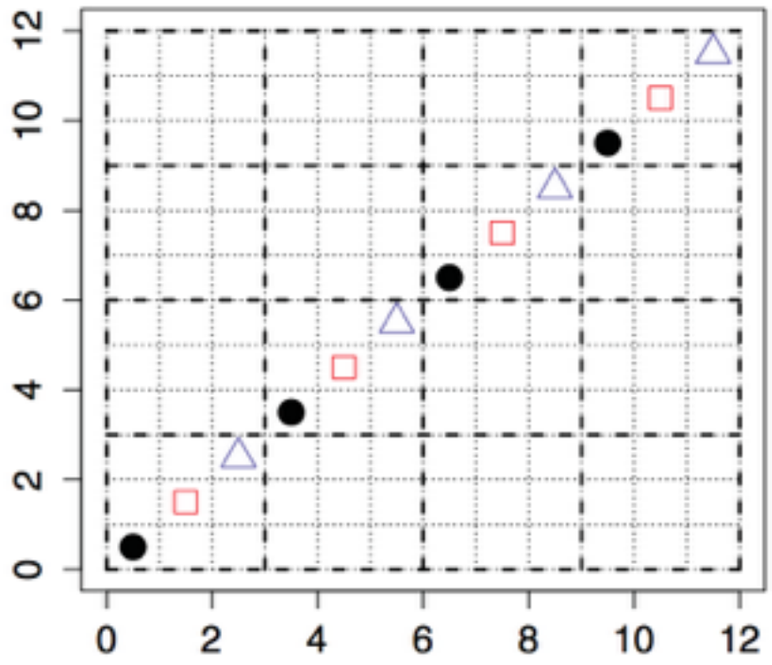
Dark Emulator Project

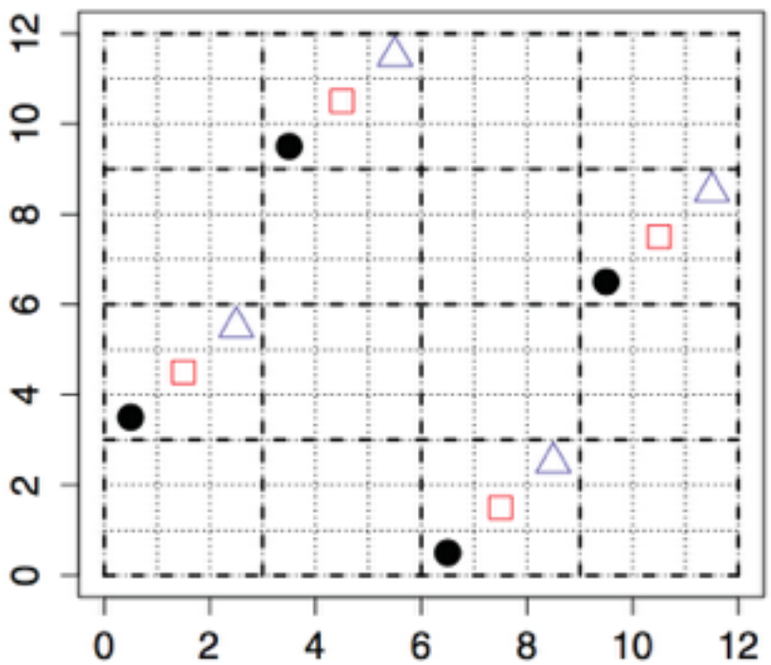
Optimal Sliced Latin Hypercube Design

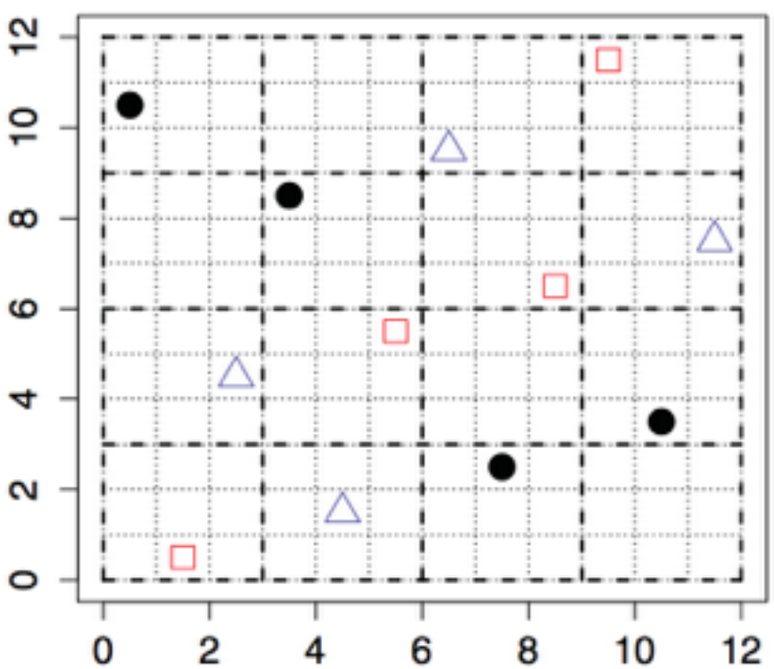
(S. Ba et al, 2015)

- パラメータ空間から偏りなく効率的にサンプリング
- 最近傍点との距離の総和を最大化するように各パラメータの値を組み合わせる







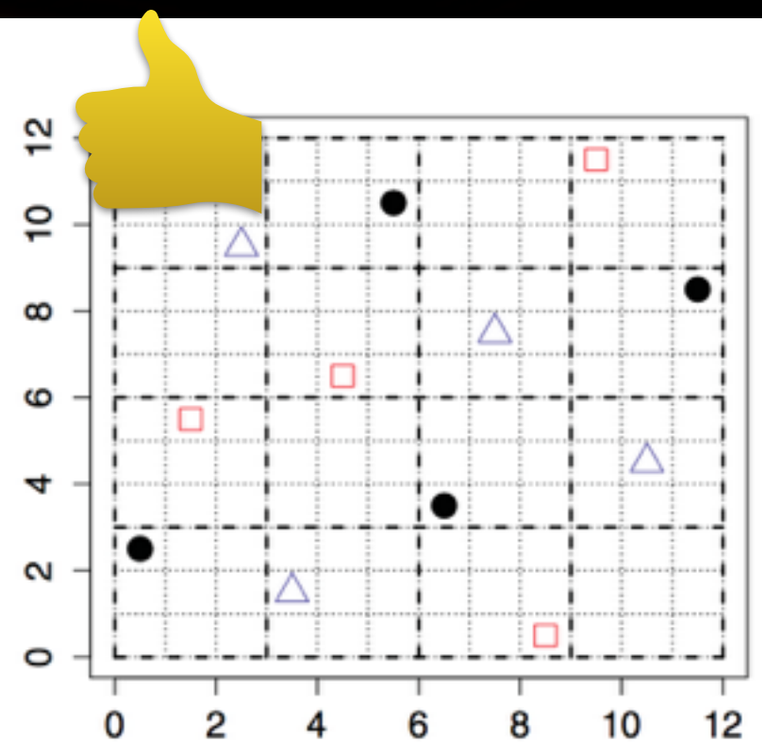


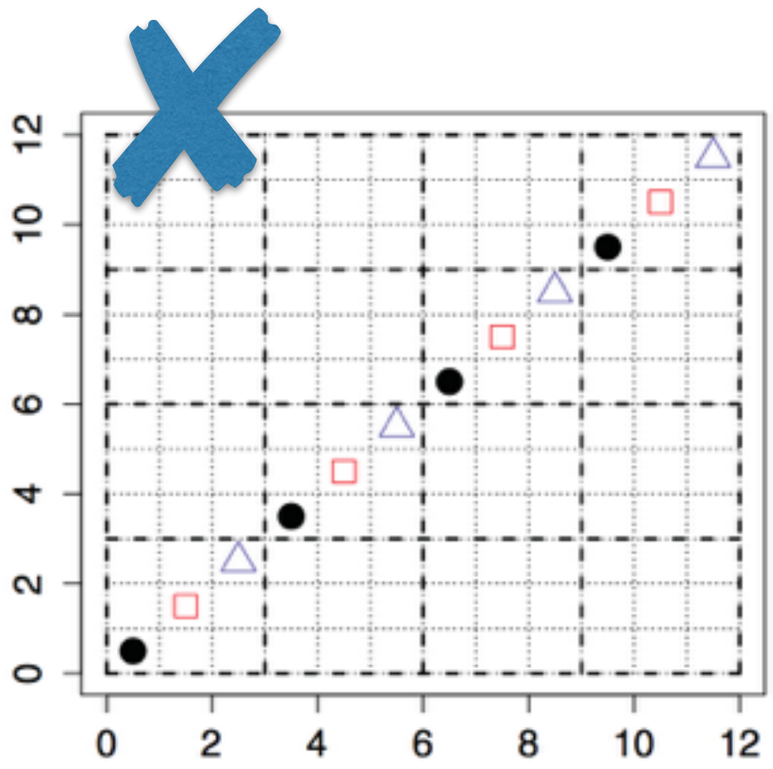
Dark Emulator Project

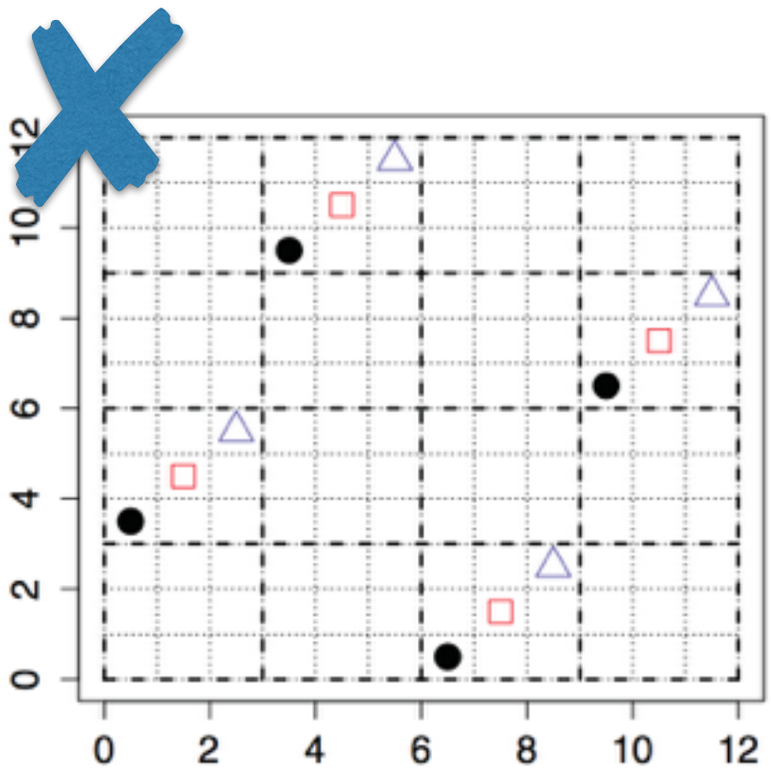
Optimal Sliced Latin Hypercube Design

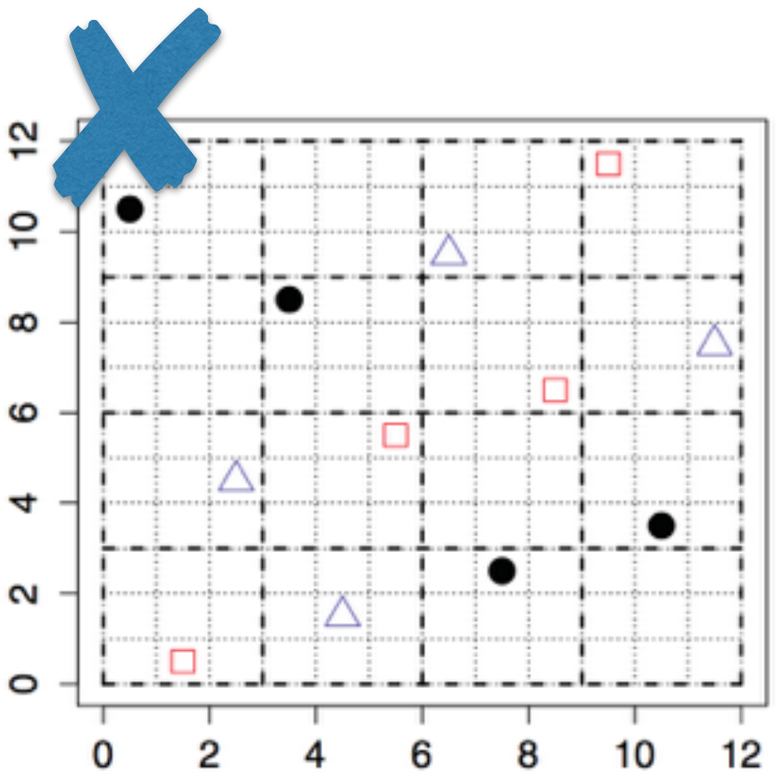
(S. Ba et al, 2015)

- パラメータ空間から偏りなく効率的にサンプリング
- 最近傍点との距離の総和を最大化するように各パラメータの値を組み合わせる







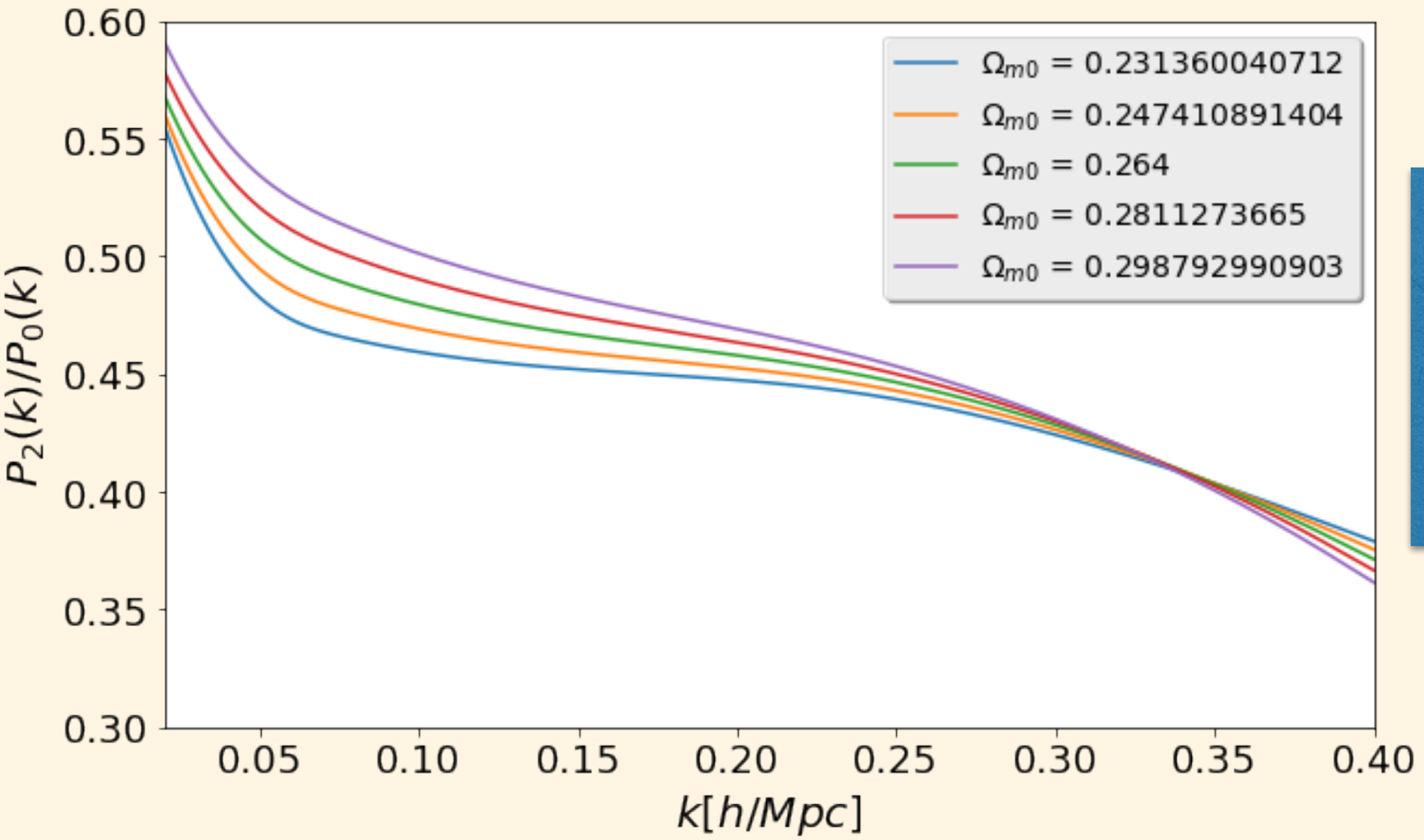


Halo Power Spectrum Emulator

研究

… a part of ‘‘Dark Emulator Project’’

- **ハローの P0(k), P2(k) の**
  宇宙論パラメータ＋赤方偏移＋ハローの質量閾値
  に対する依存性を予言する模型（エミュレータ）を構築する



宇宙論パラメータを変えても
シミュレーションし直すこと
なくパワースペクトルの理論
値を計算できる

Gaussian Process Regression

ガウス過程 ... 確率変数の集合、そのうちどの有限個をとってもガウス分布

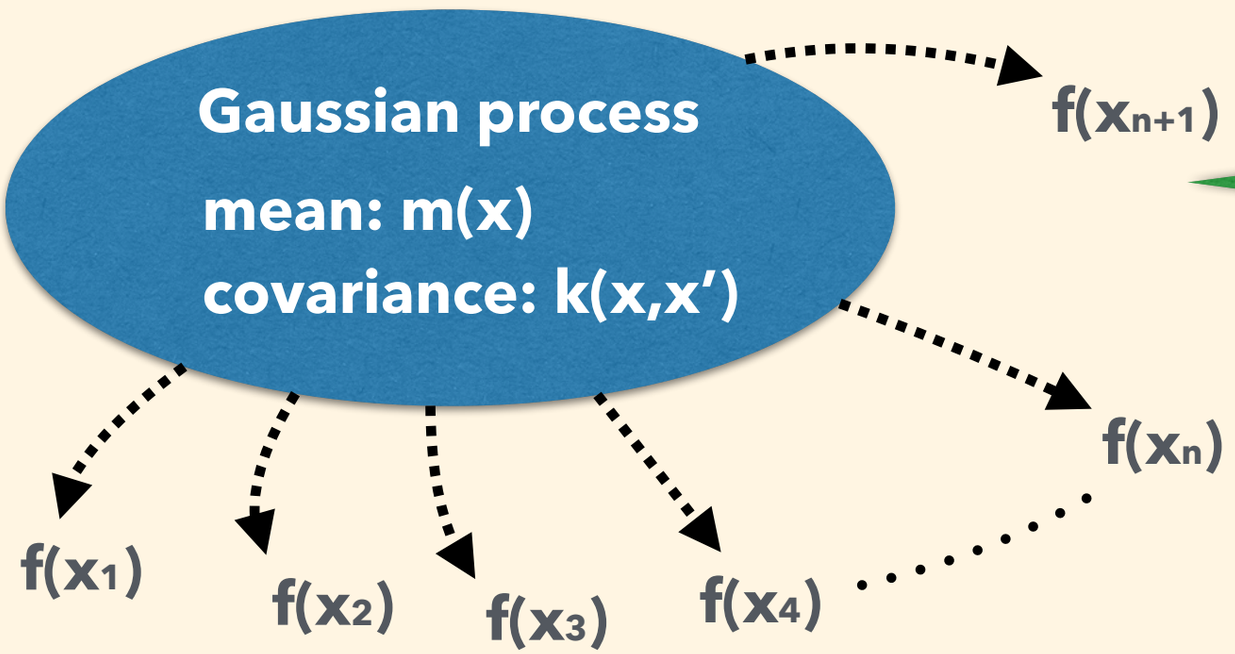
cf. Rasmussen & Williams (2006)

- 入力ベクトルとそれに対する出力値が与えられたデータ

$$\boldsymbol{x}_n \longrightarrow f(\boldsymbol{x}_n)$$

- データの出力 $f(\boldsymbol{x}_n)$ を多変量ガウス分布に従う確率変数と見なす

➡ 出力値の集まり $\{f(\boldsymbol{x}_n)\}$ が**ガウス過程**になっていると思う



n個のデータの傾向から
ガウス過程を最適化（＝学習）
n+1個目の値を予測

Gaussian Process for Power Spectrum

input

7D パラメータ $(\boldsymbol{p}_{cosmo}, z)$

$$\boldsymbol{p}_{cosmo} = (\Omega_b h^2, \Omega_c h^2, \Omega_\Lambda, \ln(10^{10} A_s), n_s, w_{de})$$

σ_8

cosmology 60 × **redshift 21**

parameter range

$0.0211 \leq \Omega_b h^2 \leq 0.0234$	$0.108 \leq \Omega_c h^2 \leq 0.132$
$0.554 \leq \Omega_\Lambda \leq 0.82$	$2.5 \leq ln(10^{10} A_s) \leq 3.7$
$0.916 \leq n_s \leq 1.012$	$-1.194 \leq w_{de} \leq -0.802$

Gaussian Process for Power Spectrum

input

7D パラメータ $(\boldsymbol{p}_{cosmo}, z)$

σ_8

$$\boldsymbol{p}_{cosmo} = (\Omega_b h^2, \Omega_c h^2, \Omega_\Lambda, \ln(10^{10} A_s), n_s, w_{de})$$

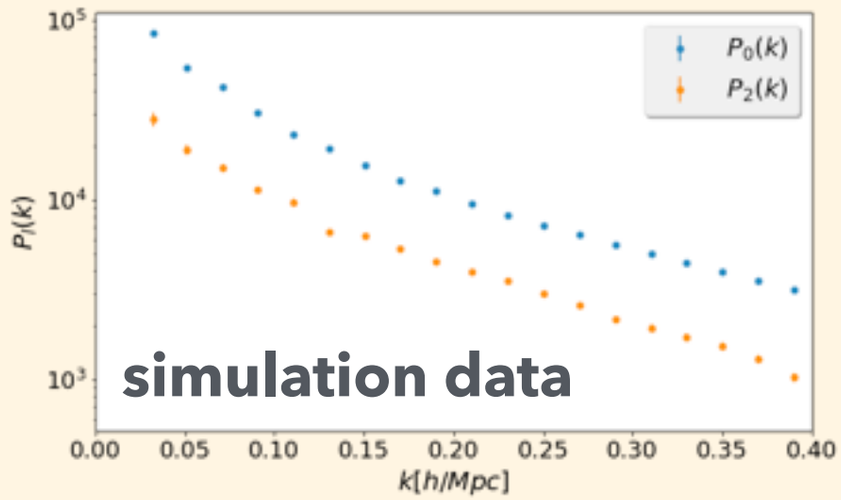
cosmology 60 × **redshift 21**

parameter range

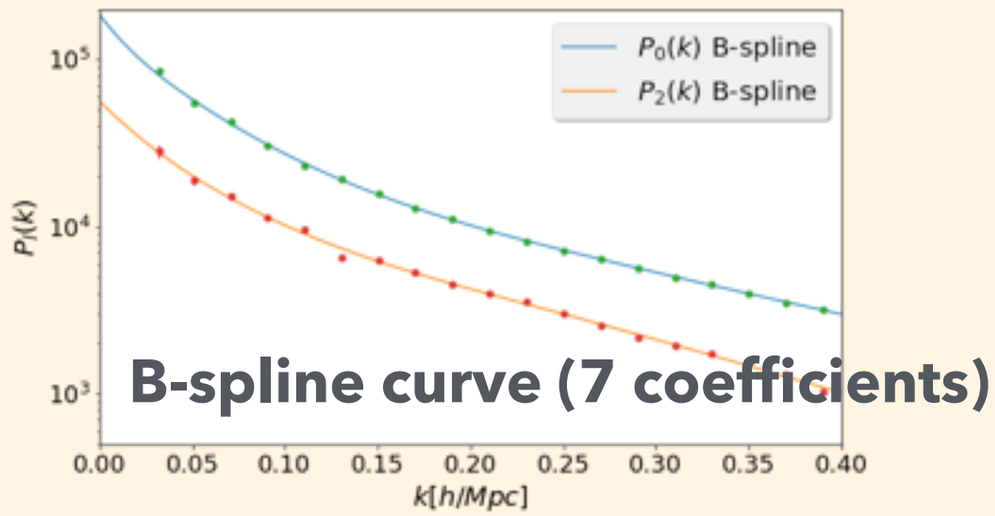
$0.0211 \leq \Omega_b h^2 \leq 0.0234$	$0.108 \leq \Omega_c h^2 \leq 0.132$
$0.554 \leq \Omega_\Lambda \leq 0.82$	$2.5 \leq ln(10^{10} A_s) \leq 3.7$
$0.916 \leq n_s \leq 1.012$	$-1.194 \leq w_{de} \leq -0.802$

output

P0(k), P2(k) をB-spline fit、各係数に対して個別にガウス過程を作って学習



次元削減



Performance of the Emulator

- シミュレーションデータをガウス過程に基づいて学習、Pythonモジュール化
- インスタンスを作ると、学習で得たデータを読み込んで学習済みのガウス過程を構成する
- 宇宙論パラメータと赤方偏移を指定すると、P0, P2 の値を高速（〜1 msec）に出力できる

jupyter halo_power_emulator Last Checkpoint: 3 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Code

In [1]:
```
%pylab inline
import sys
sys.path.append('/Users/yosuke.kobayashi/rsd/Emulator/halo_power/python')
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
from halo_power import P_l_emu
```

In [3]:
```
p0 = P_l_emu(0)
p2 = P_l_emu(2)
```

In [4]:
```
cparam = np.array([0.02225,0.1198,0.6844,3.094,0.9645,-1])
z = 0.57
k = [0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40]
```

In [5]:
```
p0(cparam, z, k)
```

Out[5]:
```
array([ 24595.28342627,  11148.09694812,   6124.10909976,   3888.46305176,
         2721.4822656 ,   2006.58288423,   1526.55430547,   1192.15457141])
```

In [6]:
```
p2(cparam, z, k)
```

Out[6]:
```
array([ 13429.65726745,   5826.04804862,   3179.51523164,   2015.38609347,
         1387.91107644,    991.09468541,    722.59774346,    531.04596224])
```

jupyter halo_power_emulator Last Checkpoint: 3 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Code

モジュールのインポート

In [1]:
```
%pylab inline
import sys
sys.path.append('/Users/yosuke.kobaya[illegible]i/rsd/Emulator/halo_power/python')
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
from halo_power import P_l_emu
```

In [3]:
```
p0 = P_l_emu(0)
p2 = P_l_emu(2)
```

In [4]:
```
cparam = np.array([0.02225,0.1198,0.6844,3.094,0.9645,-1])
z = 0.57
k = [0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40]
```

In [5]:
```
p0(cparam, z, k)
```

Out[5]:
```
array([ 24595.28342627,  11148.09694812,   6124.10909976,   3888.46305176,
         2721.4822656 ,   2006.58288423,   1526.55430547,   1192.15457141])
```

In [6]:
```
p2(cparam, z, k)
```

Out[6]:
```
array([ 13429.65726745,   5826.04804862,   3179.51523164,   2015.38609347,
         1387.91107644,    991.09468541,    722.59774346,    531.04596224])
```

jupyter halo_power_emulator Last Checkpoint: 3 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Code

モジュールのインポート

エミュレータクラスの
インスタンス

In [1]:
```
%pylab inline
import sys
sys.path.append('/Users/yosuke.kobaya[illegible]i/rsd/Emulator/halo_power/python')
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
from halo_power import P_l_emu
```

In [3]:
```
p0 = P_l_emu(0)
p2 = P_l_emu(2)
```

In [4]:
```
cparam = np.array([0.02225,0.1198,0.6844,3.094,0.9645,-1])
z = 0.57
k = [0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40]
```

In [5]:
```
p0(cparam, z, k)
```

Out[5]:
```
array([ 24595.28342627,  11148.09694812,   6124.10909976,   3888.46305176,
         2721.4822656 ,   2006.58288423,   1526.55430547,   1192.15457141])
```

In [6]:
```
p2(cparam, z, k)
```

Out[6]:
```
array([ 13429.65726745,   5826.04804862,   3179.51523164,   2015.38609347,
         1387.91107644,    991.09468541,    722.59774346,    531.04596224])
```

jupyter halo_power_emulator Last Checkpoint: 3 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Code

モジュールのインポート

エミュレータクラスの
インスタンス

宇宙論パラメータ、z、k

In [1]:
```
%pylab inline
import sys
sys.path.append('/Users/yosuke.kobayashi/rsd/Emulator/halo_power/python')
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
from halo_power import P_l_emu
```

In [3]:
```
p0 = P_l_emu(0)
p2 = P_l_emu(2)
```

In [4]:
```
cparam = np.array([0.02225,0.1198,0.6844,3.094,0.9645,-1])
z = 0.57
k = [0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40]
```

In [5]:
```
p0(cparam, z, k)
```

Out[5]:
```
array([ 24595.28342627,  11148.09694812,   6124.10909976,   3888.46305176,
         2721.4822656 ,   2006.58288423,   1526.55430547,   1192.15457141])
```

In [6]:
```
p2(cparam, z, k)
```

Out[6]:
```
array([ 13429.65726745,   5826.04804862,   3179.51523164,   2015.38609347,
         1387.91107644,    991.09468541,    722.59774346,    531.04596224])
```

jupyter halo_power_emulator Last Checkpoint: 3 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Code

モジュールのインポート

```
In [1]: %pylab inline
        import sys
        sys.path.append('/Users/yosuke.kobaya[illegible]i/rsd/Emulator/halo_power/python')
```

```
Populating the interactive namespace from numpy and matplotlib
```

エミュレータクラスの
インスタンス

```
In [2]: from halo_power import P_l_emu
```

```
In [3]: p0 = P_l_emu(0)
        p2 = P_l_emu(2)
```

```
In [4]: cparam = np.array([0.02225,0.1198,0.6844,3.094,0.9645,-1])
        z = 0.57
        k = [0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40]
```

宇宙論パラメータ、z、k

```
In [5]: p0(cparam, z, k)
```

```
Out[5]: array([ 24595.28342627,  11148.09694812,   6124.10909976,   3888.46305176,
                 2721.4822656 ,   2006.58288423,   1526.55430547,   1192.15457141])
```

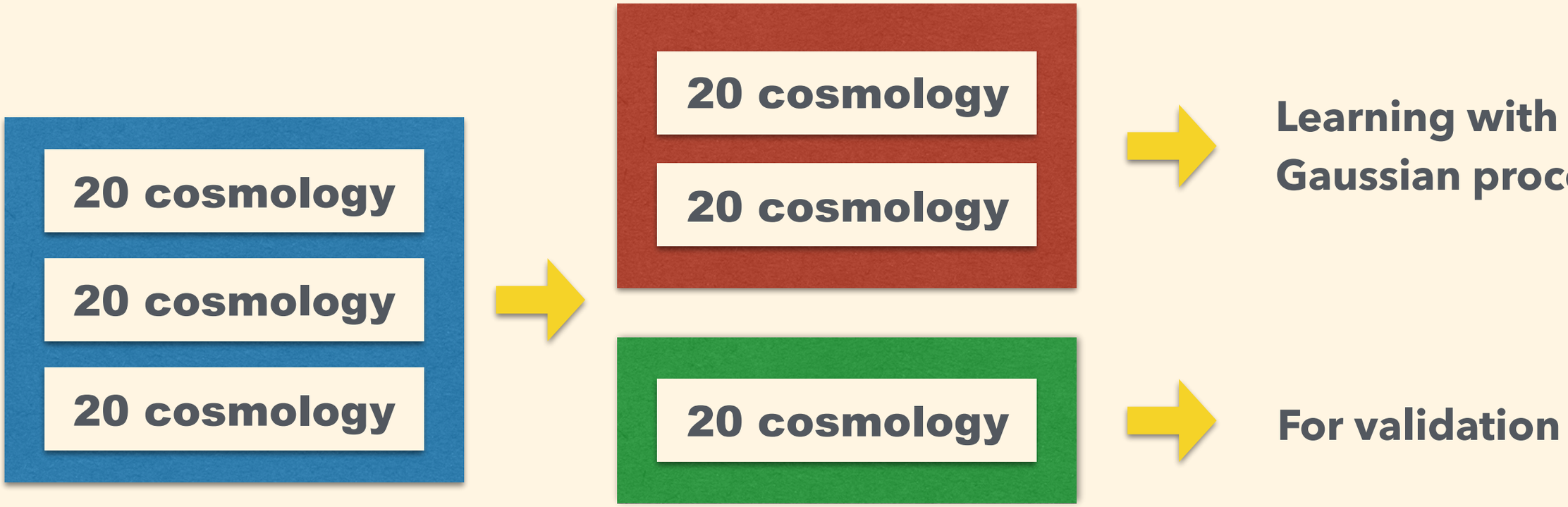
P0, P2 を出力（〜1 msec）

```
In [6]: p2(cparam, z, k)
```

```
Out[6]: array([ 13429.65726745,   5826.04804862,   3179.51523164,   2015.38609347,
                 1387.91107644,    991.09468541,    722.59774346,    531.04596224])
```

Validation of the Emulator

- 宇宙論60個を学習用と検証用に分割



- Optimal sliced Latin hypercube design で20個ずつが偏りのないサンプルセット

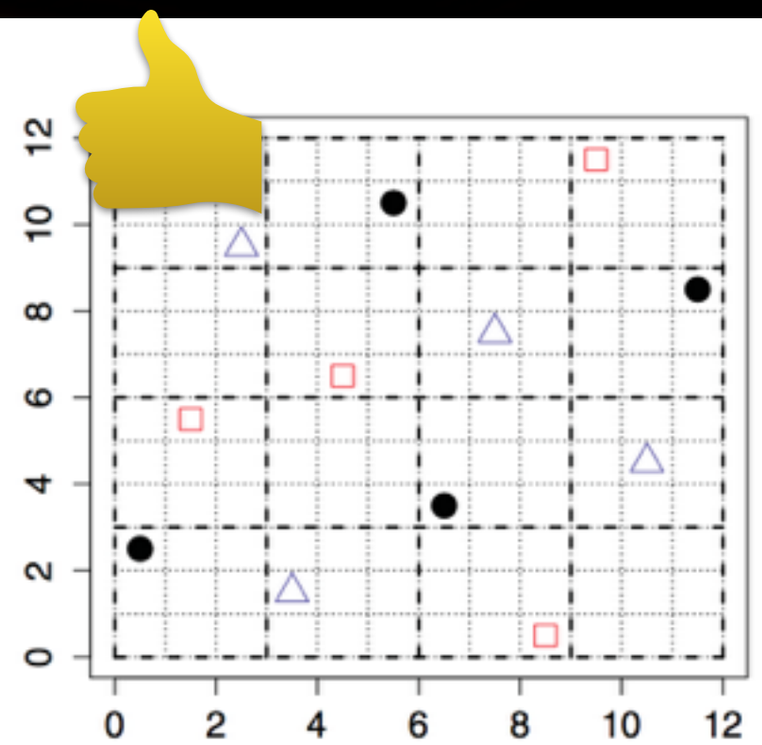
- 40個の宇宙論を学習したガウス過程が、学習に使っていない20個の宇宙論のパワースペクトルを正しく予言するか？（過学習が起こっていないか？）

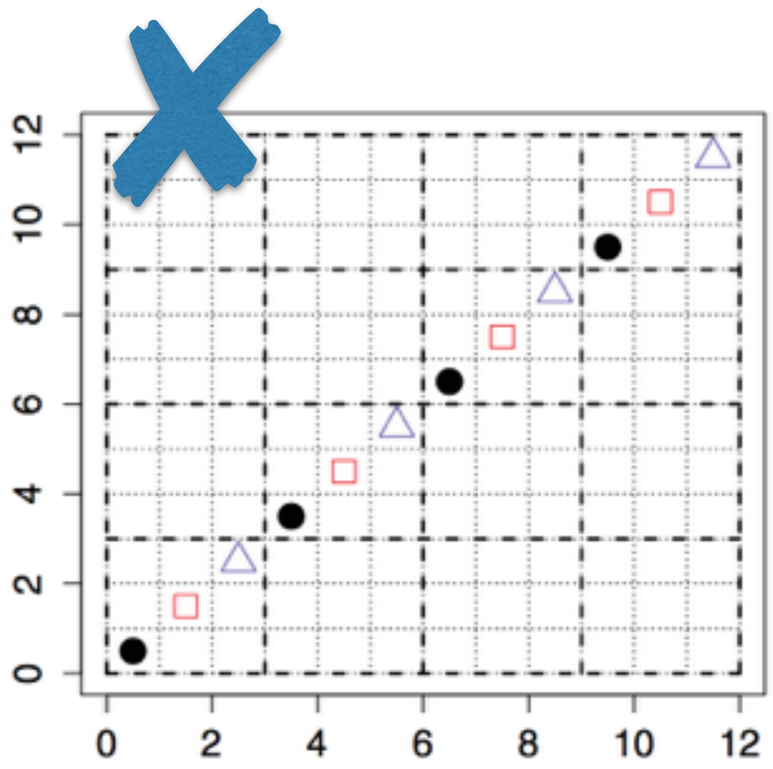
Dark Emulator Project

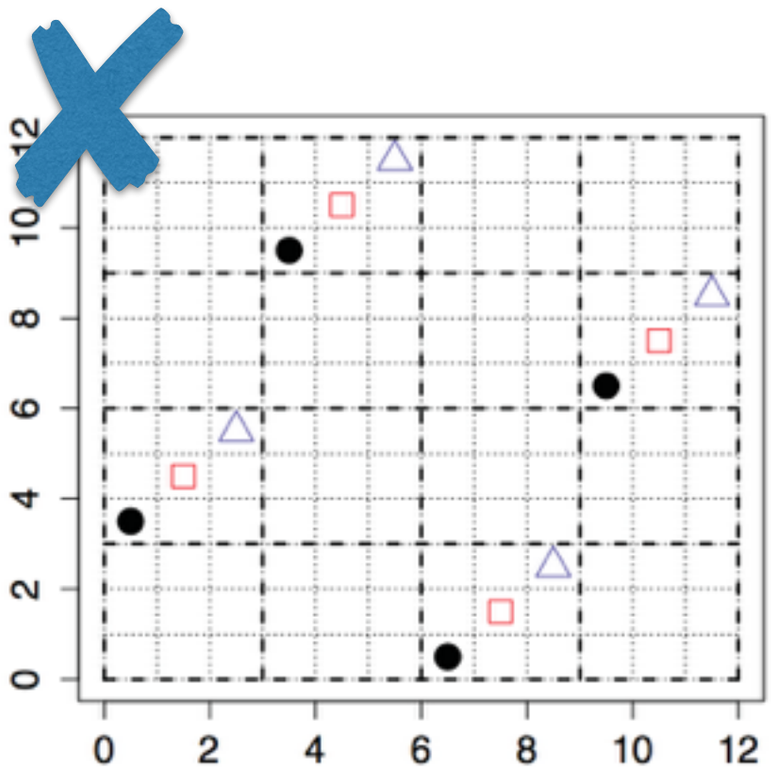
Optimal Sliced Latin Hypercube Design

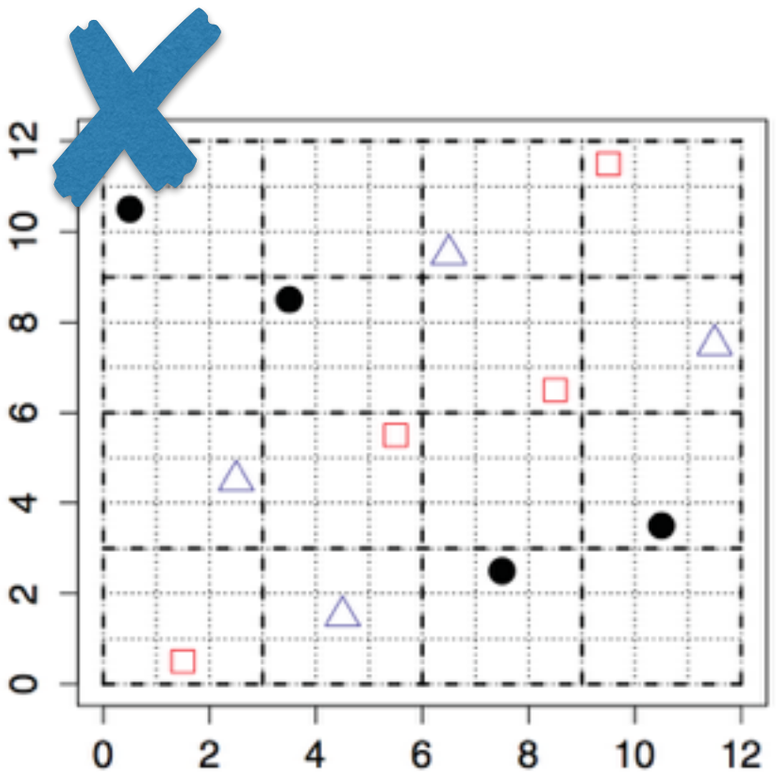
(S. Ba et al, 2015)

- パラメータ空間から偏りなく効率的にサンプリング
- 最近傍点との距離の総和を最大化するように各パラメータの値を組み合わせる









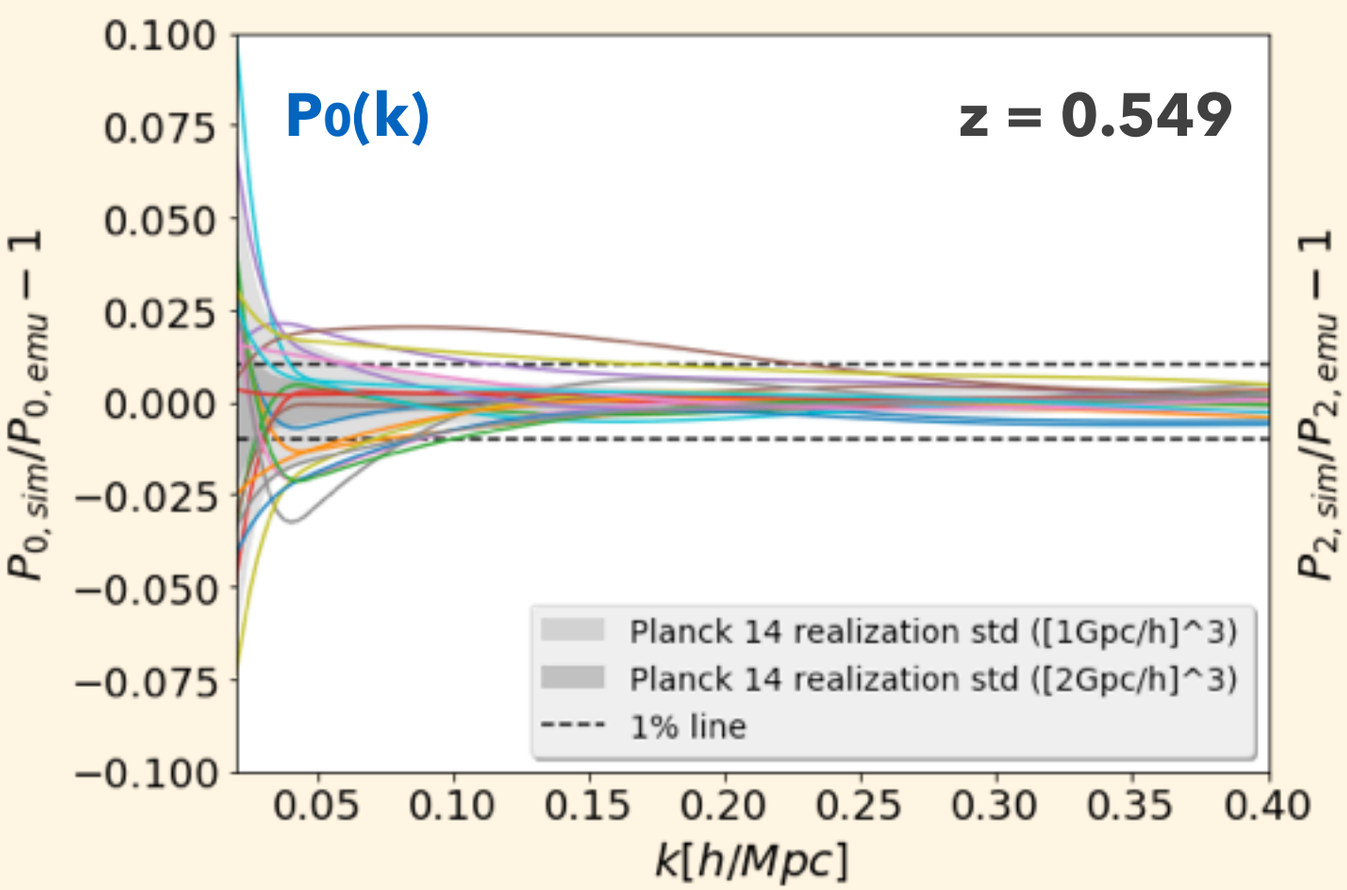
Validation of the Emulator

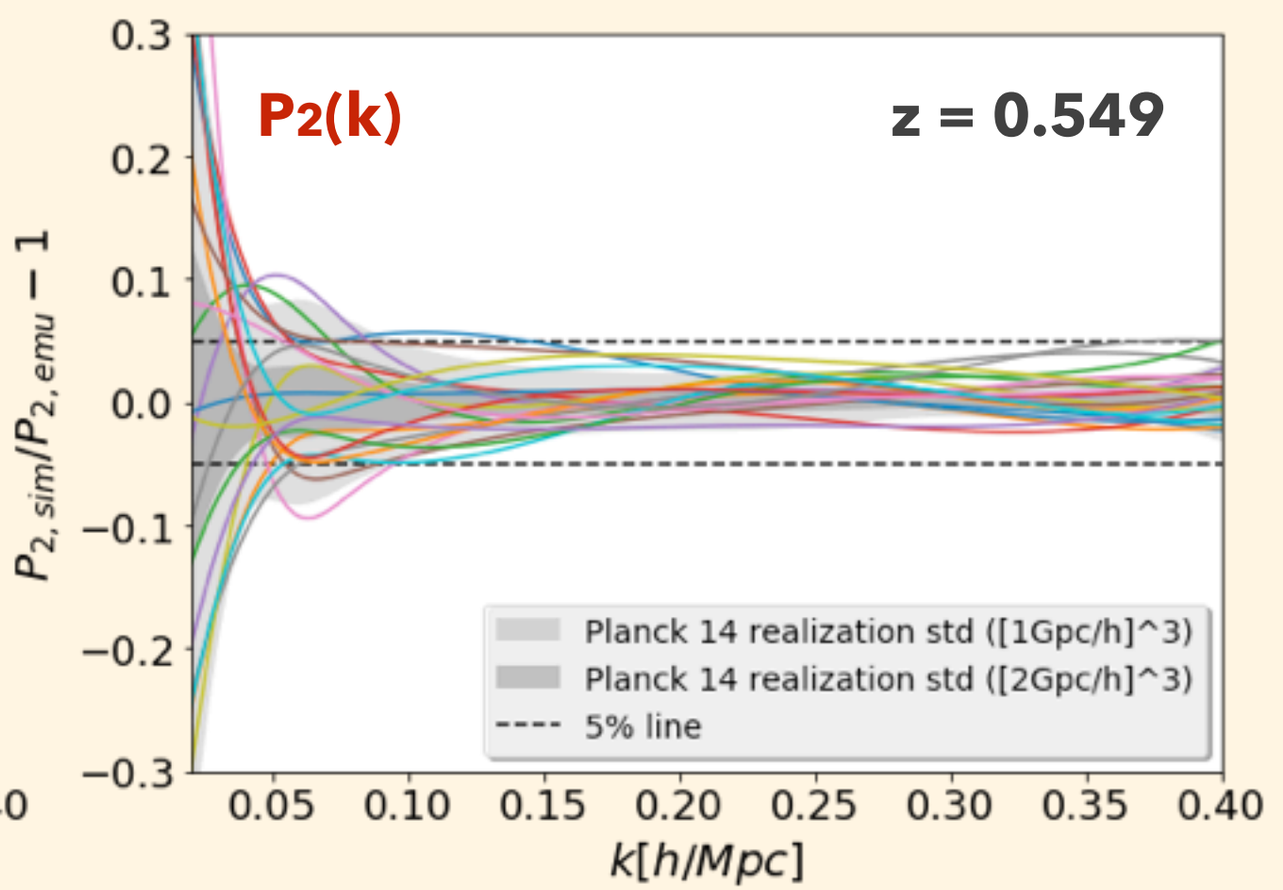
- 検証用の20宇宙論それぞれについて

$$\frac{\text{シミュレーションデータ(+ B-spline fit)}}{\text{40宇宙論を学習して作ったエミュレータによる予言}} - 1$$

を計算した結果

$n_{\rm halo} = 10^{-3}[h/{\rm Mpc}]^3$ ~ WFIRST galaxy number density





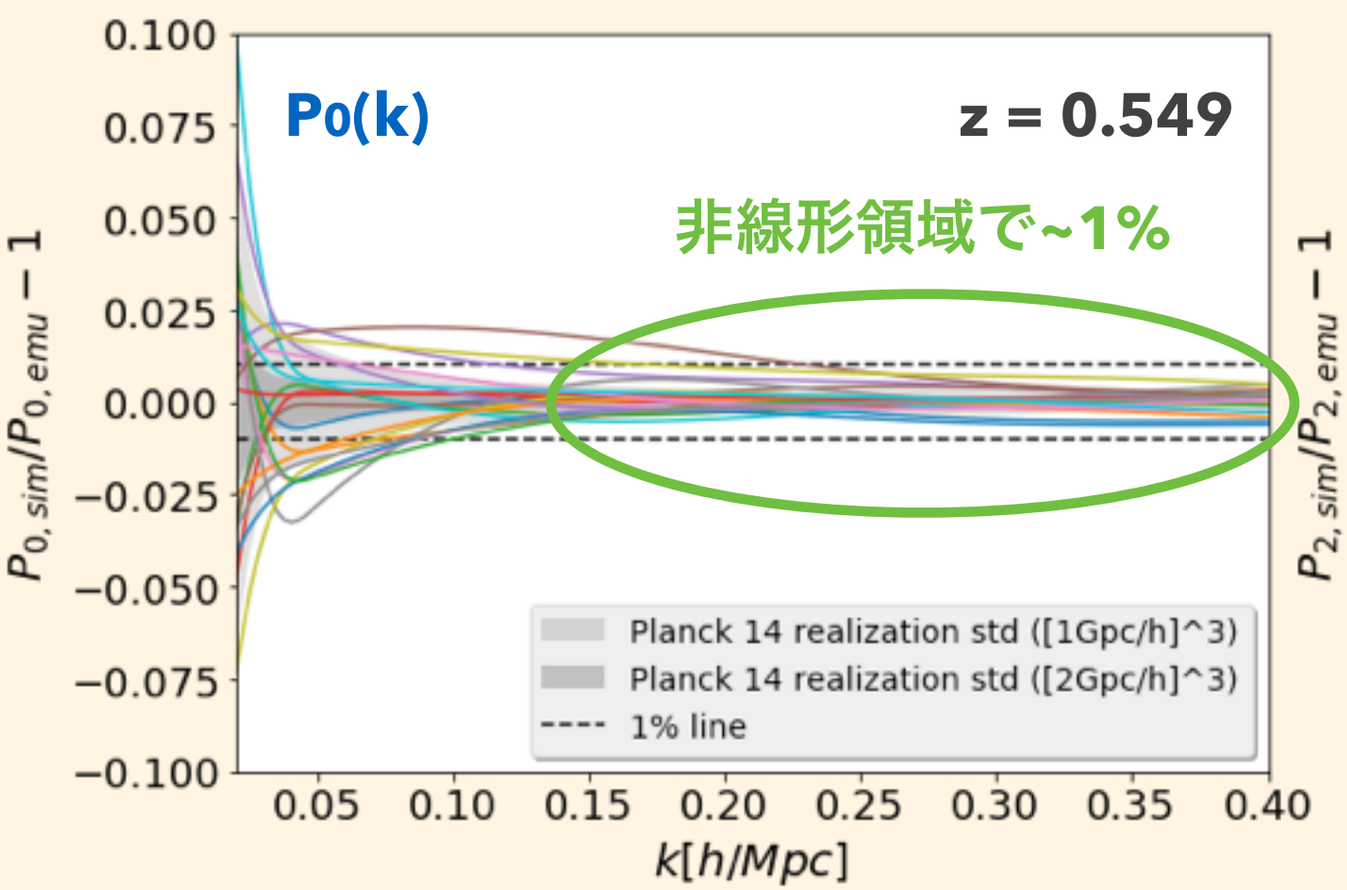
Validation of the Emulator

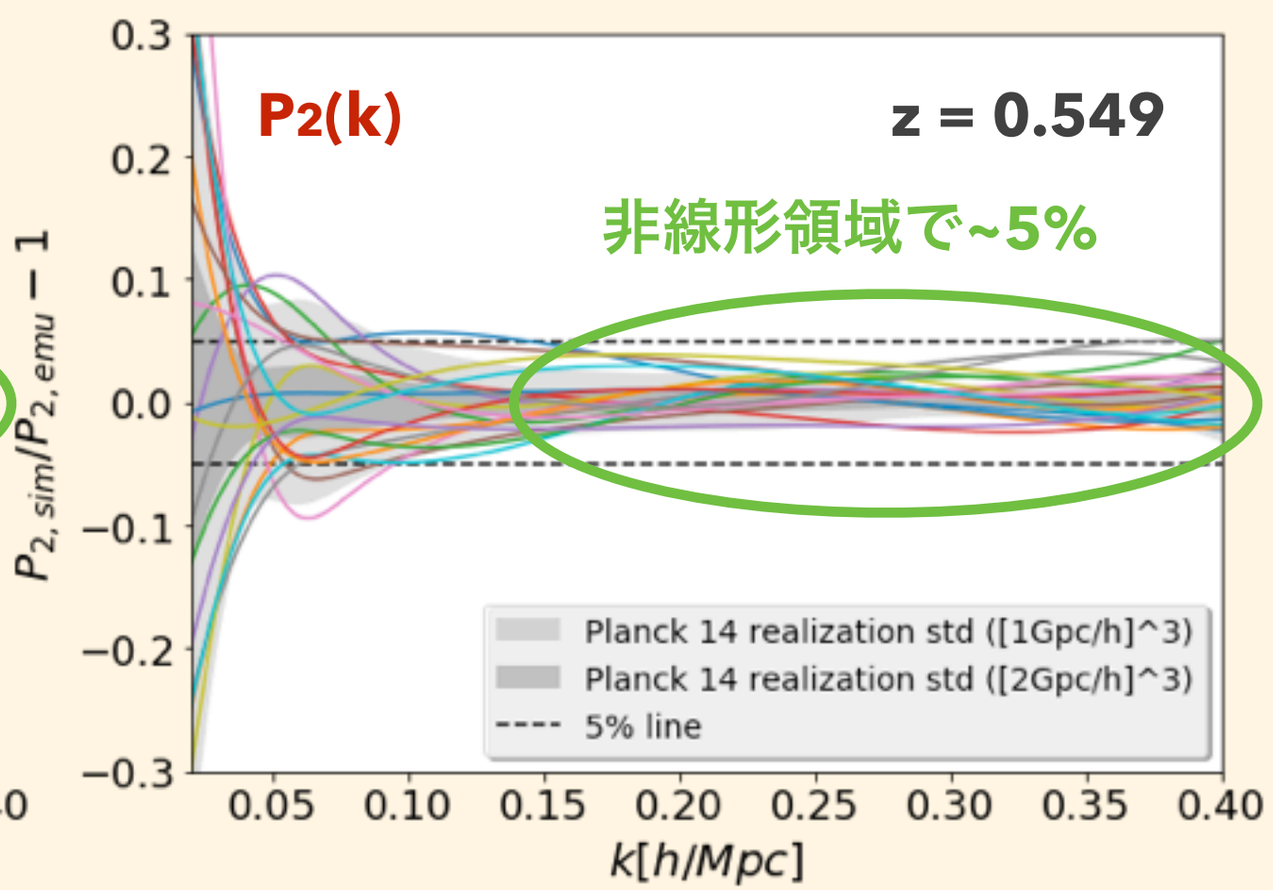
- 検証用の20宇宙論それぞれについて

$$\frac{\text{シミュレーションデータ(+ B-spline fit)}}{\text{40宇宙論を学習して作ったエミュレータによる予言}} - 1$$

を計算した結果

$n_{\rm halo} = 10^{-3}[h/{\rm Mpc}]^3$ ~ WFIRST galaxy number density





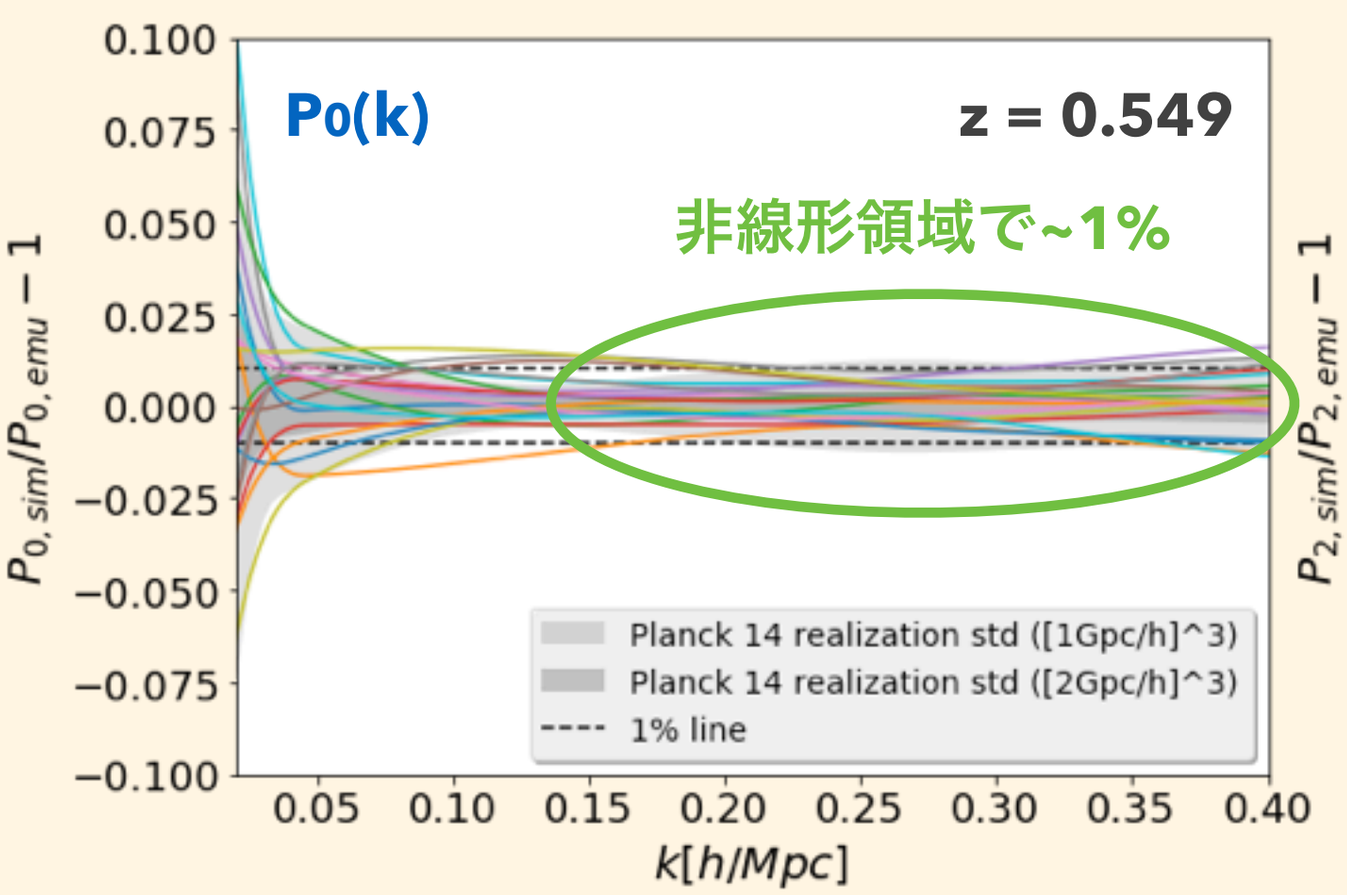
Validation of the Emulator

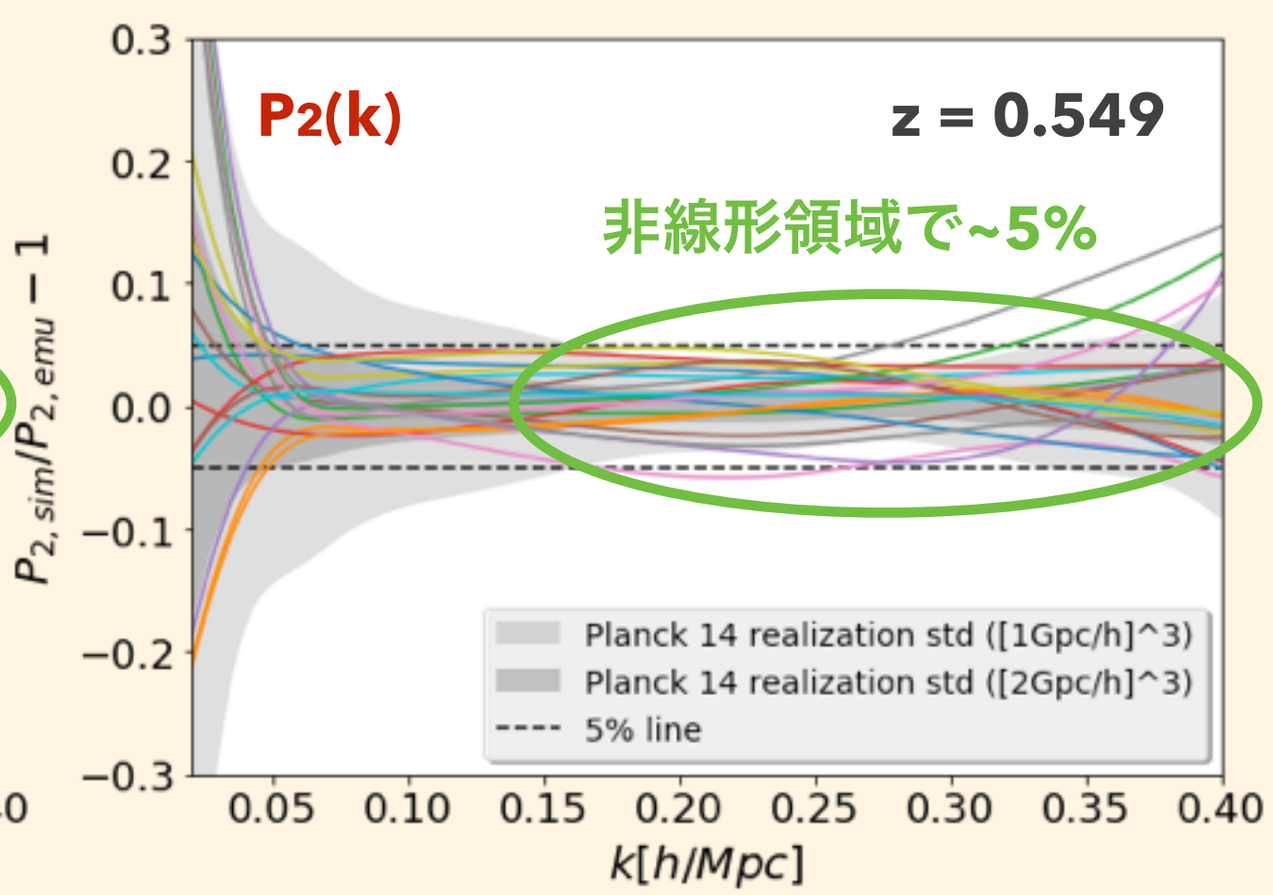
- 検証用の20宇宙論それぞれについて

$$\frac{\text{シミュレーションデータ(+ B-spline fit)}}{\text{40宇宙論を学習して作ったエミュレータによる予言}} - 1$$

を計算した結果

$n_{\rm halo} = 10^{-4}[h/{\rm Mpc}]^3$ ~ BOSS galaxy number density





Conclusion & Future Work

- **B-spline fit + ガウス過程で、赤方偏移空間におけるハローのパワースペクトルのP0, P2をそれぞれ1%, 5%程度で予言できた**

- **k ~ 0.4 h/Mpc までの非線形スケールで高精度予言**

- **シミュレーションの追加による統計の増加で精度向上が期待**

Future Work

数〜数十ミリ秒単位での出力？

- **ハローの質量閾値に対する依存性の実装**

- **エミュレータを用いた Fisher forecast**