Lecture 4
Model Selection & Development

Joe Zuntz

Evidence

Model Selection

- Given two models, how can we compare them?
- Simplest approach = compare ML
  - Does not include uncertainty or Occam's Razor
- Recall that all our probabilities have been conditional on the model, as in Bayes:

$$P(p|M) = \frac{P(d|pM)P(p|M)}{P(d|M)}$$

Model Selection: Bayesian Evidence

- Can use Bayes Theorem again, on model level:

$$P(M|d) = \frac{P(d|M)P(M)}{P(d)}$$

- Only really meaningful when comparing models.
  Bayes Factor B:

$$\frac{P(M_1|d)}{P(M_2|d)} = \underbrace{\frac{P(d|M_1)}{P(d|M_2)}}_{\text{Bayesian Evidence Values}} \; \frac{P(M_1)}{P(M_2)}$$

Model Priors

Model Selection: Bayesian Evidence

- Likelihood of parameters within model:

$$P(d|pM)$$

- Evidence of model:

$$P(d|p)$$

Model Selection: Bayesian Evidence

- Evidence is the bit we ignored before when doing parameter estimation

- Given by an integral over prior space

$$P(d|M) = \int P(d|pM)P(p|M)\mathrm{d}p$$

- Hard to evaluate - posterior usually small compared to prior

Model Selection: Evidence Approximations

- Nice evidence approximations for some cases:
    - Savage-Dickey Density ratio
      (for when one model is a subset of another)
    - Akaike information criterion AIC
      Bayesian information criterion BIC
      Work in various circumstances

Savage Dickey

- Applies to two models where M_1 is restricted version of M_2
    - e.g M_1 = LCDM $\underline{\Omega}=\{\Omega_m, \Omega_b, \ldots\}$ with $w=-1$
      M_2 = wCDM

$$P(d|\underline{\Omega}, M_1) = P(d|\underline{\Omega}, w=-1, M_2)$$

    - with separable priors

$$P(\underline{\Omega}|w=-1, M_2) = P(\underline{\Omega}, M_1)$$

Savage Dickey

- In this case, Bayes factor given by

$$\frac{P(w=-1|d,M_2)}{P(w=-1|M_2)}$$

Posterior, integrated over <u>Ω</u>

Prior

Model Selection: Nested Sampling

$$\int L(\theta)p(\theta)\mathrm{d}\theta = \int L(X)\mathrm{d}X$$

$$\approx \sum L_i \delta X_i$$

$$\mathrm{d}X \equiv P(\theta)\mathrm{d}\theta$$

X = remaining prior volume

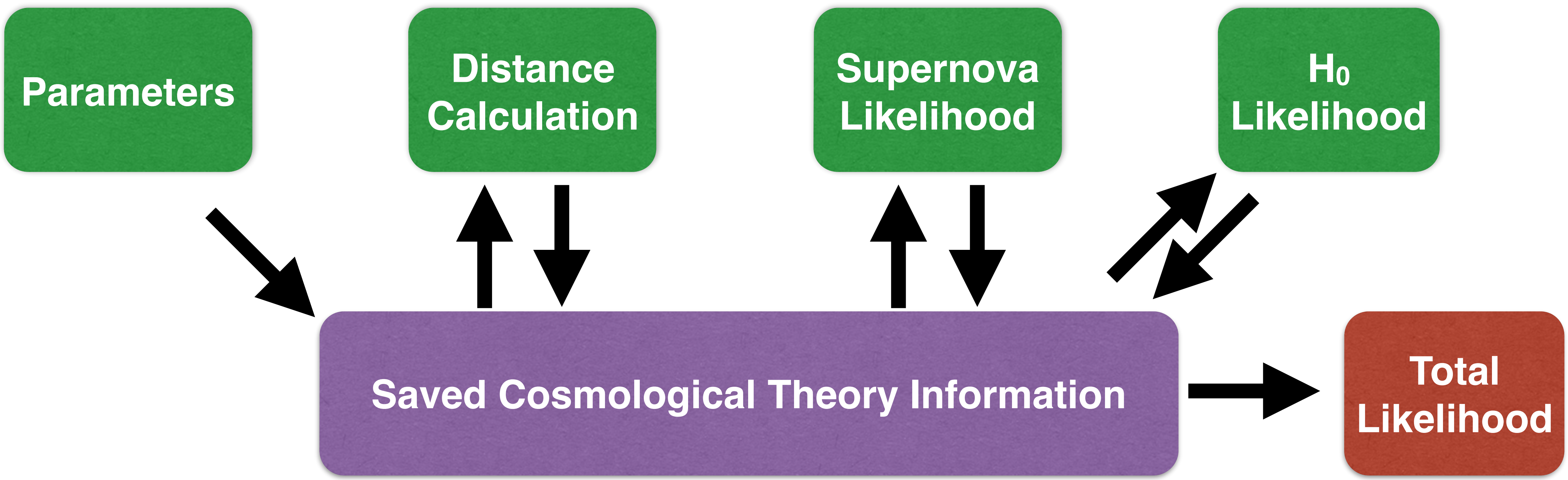
Model Selection: Nested Sampling

- Also uses ensemble of live points
  - Computes constraints as well as evidence
- Each iteration, replace lowest likelihood point with one higher up, sampled from prior
- Multinest software is extremely clever
  - C, F90, Python bindings

Multinest Example

```
> cosmosis demos/demo9.ini

> postprocess -o plots -p demo9 demos/demo9.ini
  --extra demos/extra9.py
```



https://bitbucket.org/joezuntz/cosmosis/wiki/Demo9

More Samplers

Importance Sampling

- Re-sampling from re-weighted existing samples
    - Changed prior / likelihood
    - New data

$$
\begin{aligned}
E[f(x)] &= \int P_1(x) f(x) \mathrm{d}x \approx \frac{1}{N} \sum_{\text{Chain 1}} f(x_i) \\
&= \int \left(\frac{P_1(x)}{P_2(x)} f(x) \right) P_2(x) \mathrm{d}x \approx \frac{1}{N} \sum_{\text{Chain 2}} f(x_i) \frac{P_1(x_i)}{P_2(x_i)}
\end{aligned}
$$

Importance Sampling

- i.e.
  - Take a chain you sampled from some distribution P_2
  - Give each sample a weight $P_1(x)/P_2(x)$ for some new distribution P_1
  - Make your histograms, estimates, etc, using these weights

Importance Sampling

- Works better the more similar P_2 is to P_1
- Won’t work if P_2 small where P_1 isn’t
  - So better for extra data than different data

Gibbs Sampling

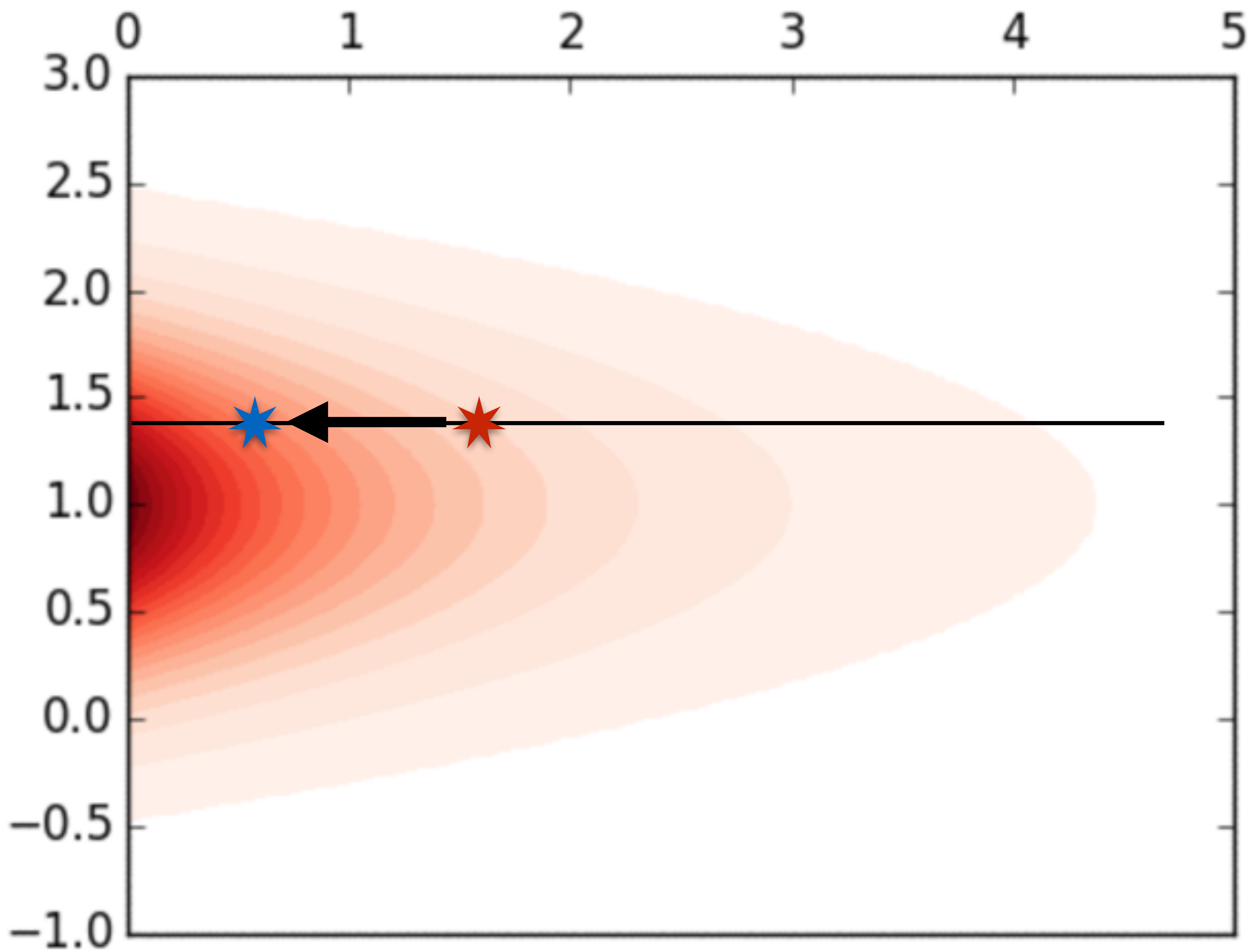
- Applicable when have >1 parameters a, b, c, … z
- And can directly sample from conditional likelihoods: $P(a|bcd\ldots)$, $P(b|acd\ldots)$, $P(c|abd\ldots)$, … $P(z|abc\ldots y)$
- Can be very efficient when possible

Gibbs Sampling

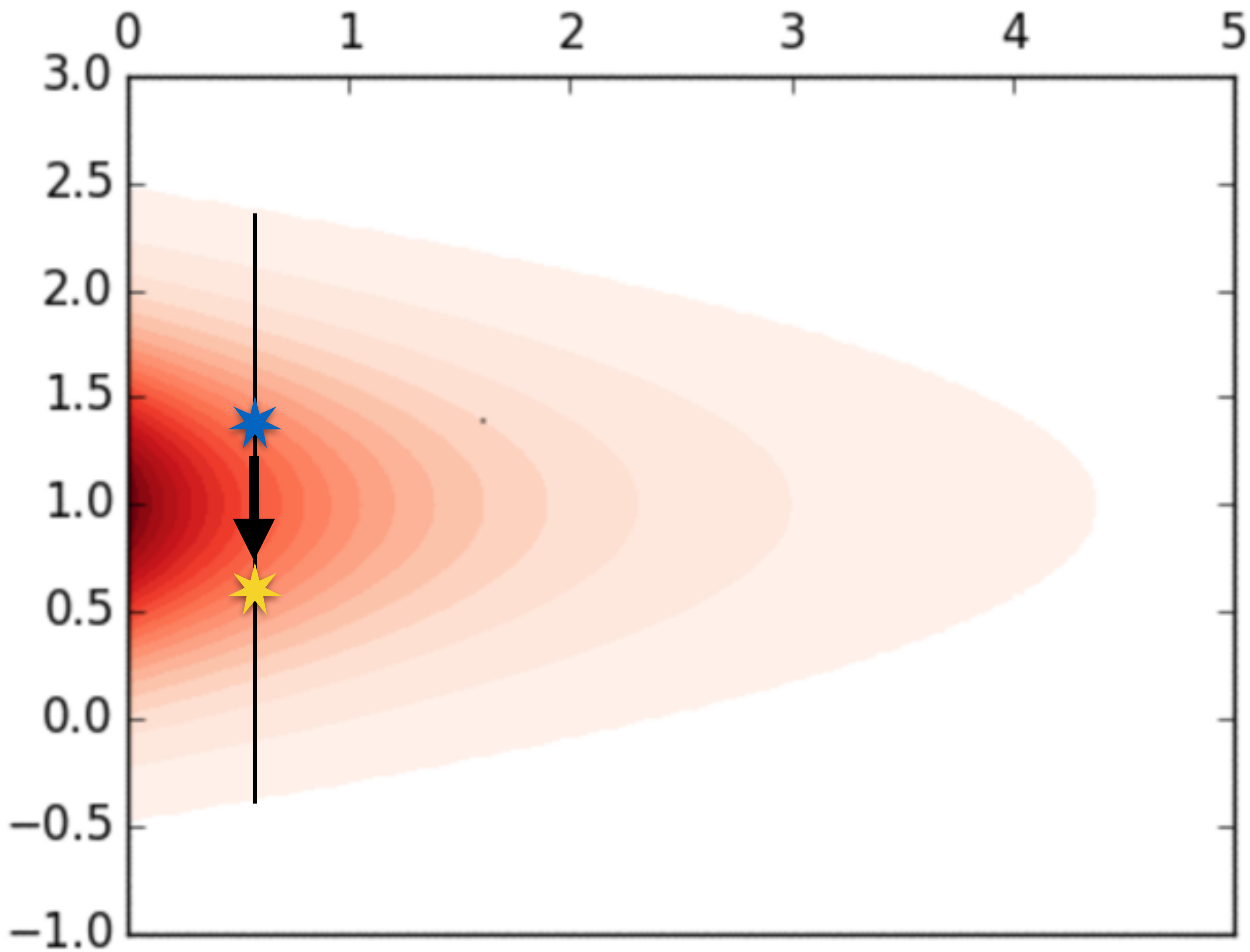
- Very simple algorithm - just each parameter in turn
- 2D version with parameters (a,b):

$$
\begin{aligned}
&\text{for } i = 1 \ldots \\
&\quad a^{i+1} \sim P(a^{i+1} | b^i) \\
&\quad b^{i+1} \sim P(b^{i+1} | a^{i+1})
\end{aligned}
$$

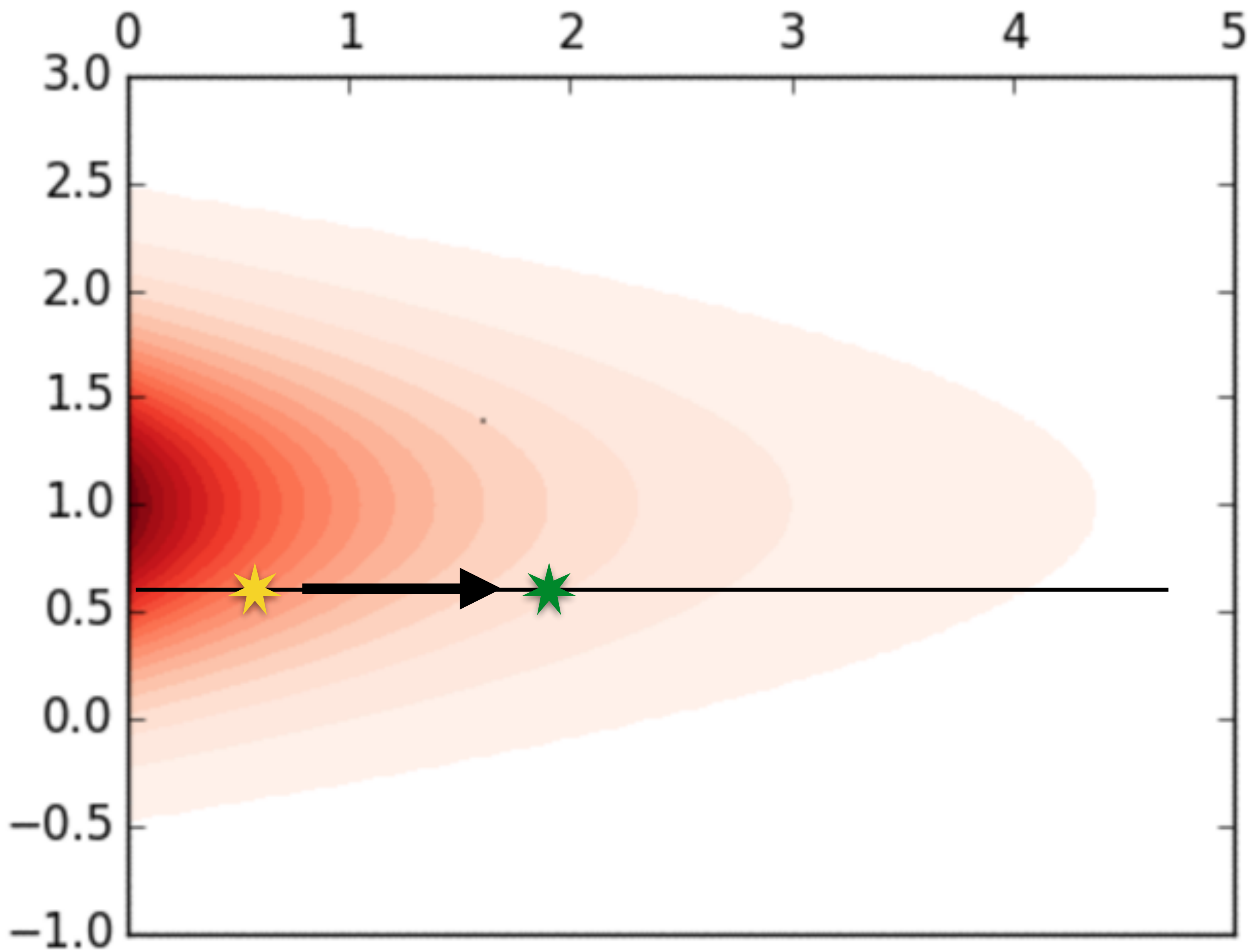
Gibbs Sampling



Gibbs Sampling



Gibbs Sampling



Gibbs Sampling

- Multi-parameter case - not as bad as it looks:

$$\text{for } i = 1...$$
$$\quad \text{for } k = 1...n_{\text{param}}$$
$$\quad\quad x_k^{i+1} \sim P(x_k^{i+1} | x_1^{i+1}, x_2^{i+1}, ..., x_{k-1}^{i+1}, x_{k+1}^{i}, x_{k+2}^{i}, ..., x_{n_{\text{param}}}^{i})$$

- Can also block groups of parameters together and update as vectors

Defining a pipeline run

Pipeline Definition

- Look at demos/demo2.ini

  ```
  [pipeline]
  modules = consistency camb planck bicep
  values = demos/values2.ini
  ```

- Each module in the list is described lower down - file path to module and any options for it

- Parameters defined in the “values” file

Building & extending likelihood pipelines

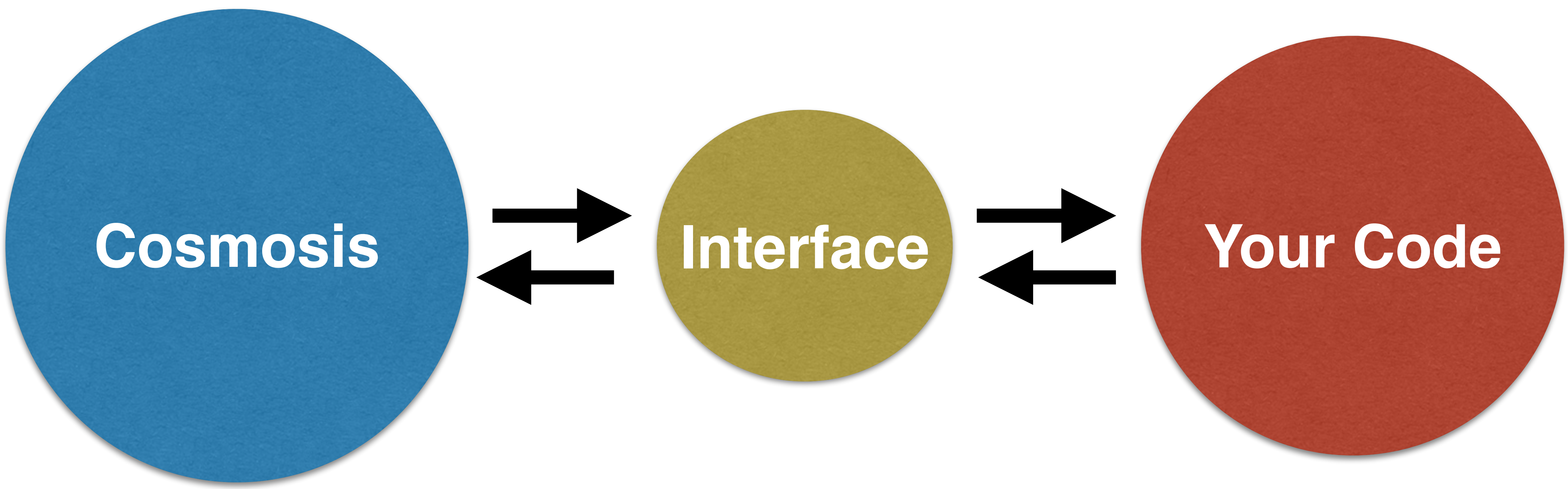
Managing Code

- *Design* before you *write*
- Read about how to code!
- If you don't use version control you are definitely making a mistake.
    - Learn *git*. It's worth it.

Organizing Likelihoods

- Separate theory calculation from likelihood
    - Can replace methods and data independently
- Don't Repeat Yourself (D.R.Y.)
    - Use existing distance calculations, P(k,z), etc.
- Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, Libraries, *Libraries*, **Libraries**, ***Libraries***.

Connecting Code



Creating a cosmosis module

- Given a piece of code implementing your module, we will write an interface connecting it to cosmosis

- Need two functions:
  setup, execute

- https://bitbucket.org/joezuntz/cosmosis/wiki/modules_python

- https://bitbucket.org/joezuntz/cosmosis/wiki/modules_c

- https://bitbucket.org/joezuntz/cosmosis/wiki/modules_fortran

Setup

- Cosmology-independent settings and setup
    - e.g. loading data, limits on
- Read settings from ini file

Execute

- Cosmology calculations
- Main module work
- Read inputs (from cosmosis)
- Save outputs (to cosmosis)

Three Groups

- Non-programmers
    - Go through the demos at https://bitbucket.org/joezuntz/cosmosis
- Did homework and coded Cepheid likelihood 😀
    - Create likelihood module
- Didn’t do homework! 😠
    - Test a new w(z) theory

Creating a Likelihood

- Last time you coded up a likelihood for the LMC and extragalactic Cepheids

- Here's some data!
  LMC http://bit.ly/1vQ4RTV
  Ex-gal http://bit.ly/1tJzRBT

- Note: there are complexities I skipped when describing this! You'll only get H_0 to a factor of a few.

- Let's turn this into a cosmosis module

- Inputs: `h0, alpha, beta` in `cosmological_parameters`

- Outputs: `cepheid_like` in `likelihoods`

Testing A New Theory

- Let's constrain the w_0-w_z parameterisation

$$w(z) = w_0 + w_z z$$

$$\Omega_\Lambda(z) = \Omega_\Lambda(0)\,(1+z)^{3(1+w_0-w_z)}\exp\left(-3w_z z\right)$$

- Use scipy.integrate.quad to do the integration

- Inputs: `h0`, `omega_m`, `w0`, `wz` in `cosmological_parameters`

- Outputs: `z`, `mu` in `distances`

Distance Equations

$$\Omega_\Lambda(z) = \Omega_\Lambda(0)\,(1+z)^{3(1+w_0-w_z)}\exp\left(-3w_z z\right)$$

$$\Omega_m(z) = \Omega_m(1+z)^3$$

$$H(z) = H_0\sqrt{\Omega_m(z) + \Omega_\Lambda(z)}$$

$$D_c(z) = c\int_0^z \frac{1}{H(z')}\mathrm{d}z'$$

$$D_L(z) = (1+z)D_c(z)$$

$$\mu(z) = 5\log_{10}\frac{D_L}{\mathrm{Mpc}} - 25$$

Example Implementation

http://nbviewer.ipython.org/gist/joezuntz/d4b82ce5b3010870aa6b

Getting Started

- Create a new directory under modules/
- Put your code in a file in there
- Create another file in there (same language) to connect to cosmosis
- See the wiki links above for examples of what they look like - adapt these for your code