

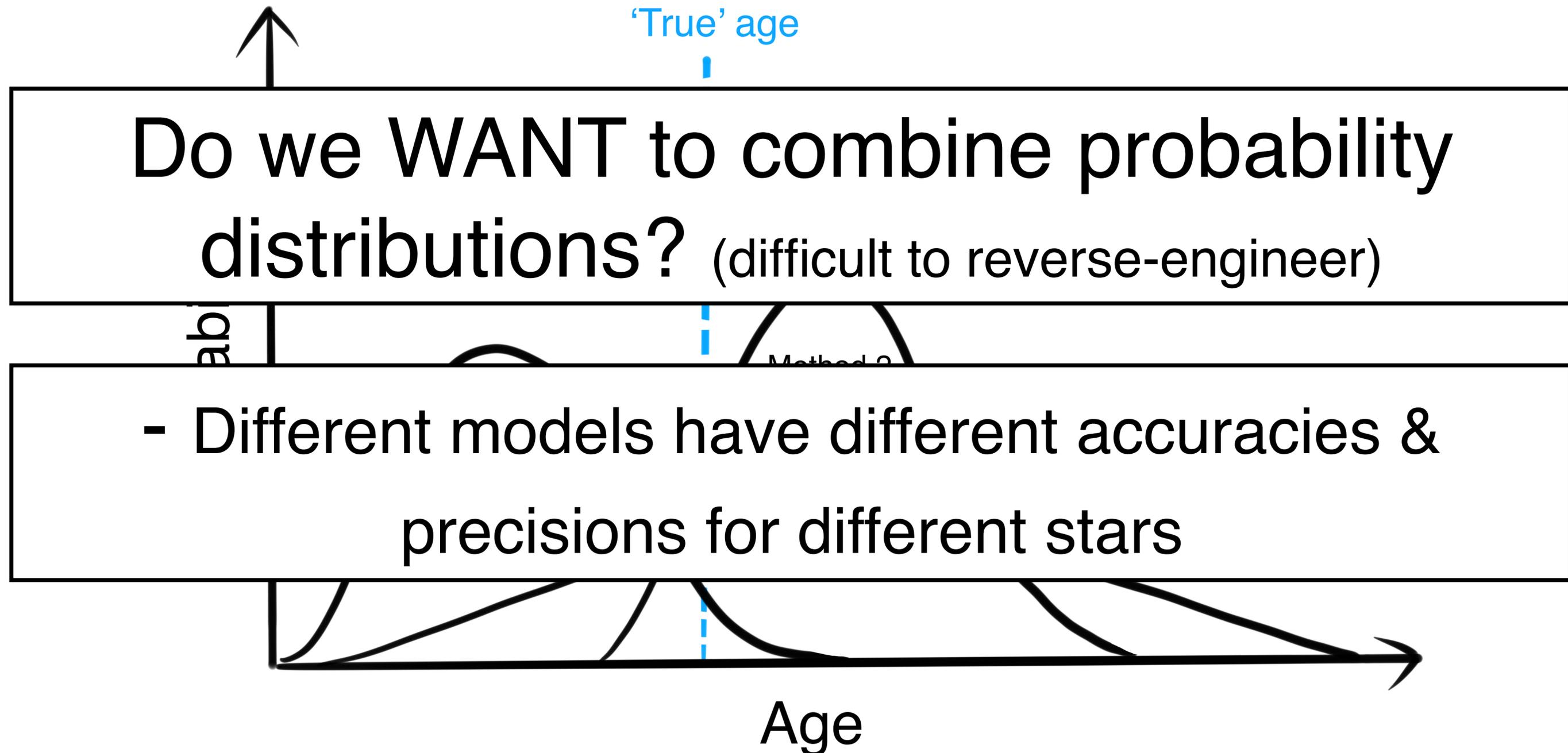
# Final stellar parameters: Combining methods

**Ruth Angus**

**Assistant Curator, Department of Astrophysics, American Museum of Natural History  
Associate Research Scientist, Center for Computational Astrophysics, Flatiron Institute  
Assistant Adjunct Professor, Department of Astronomy, Columbia University**

# What is the final age?

How do you **combine** probability distributions?



# stardate (Cross-published in AJ and JOSS)

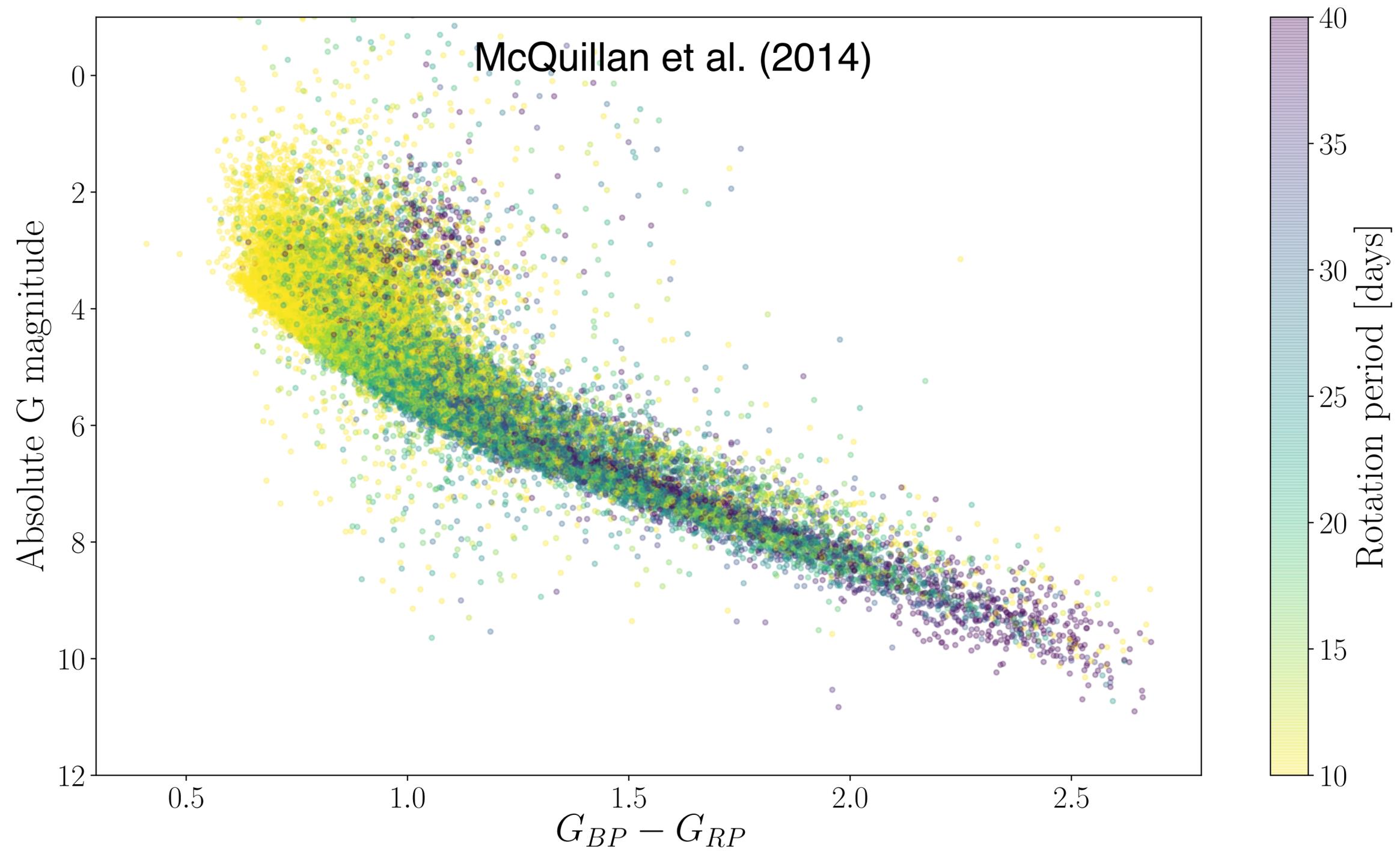
Angus et al (2019a, b)

[stardate.readthedocs.io](https://stardate.readthedocs.io)  
[github/ruthangus/stardate](https://github.com/ruthangus/stardate)

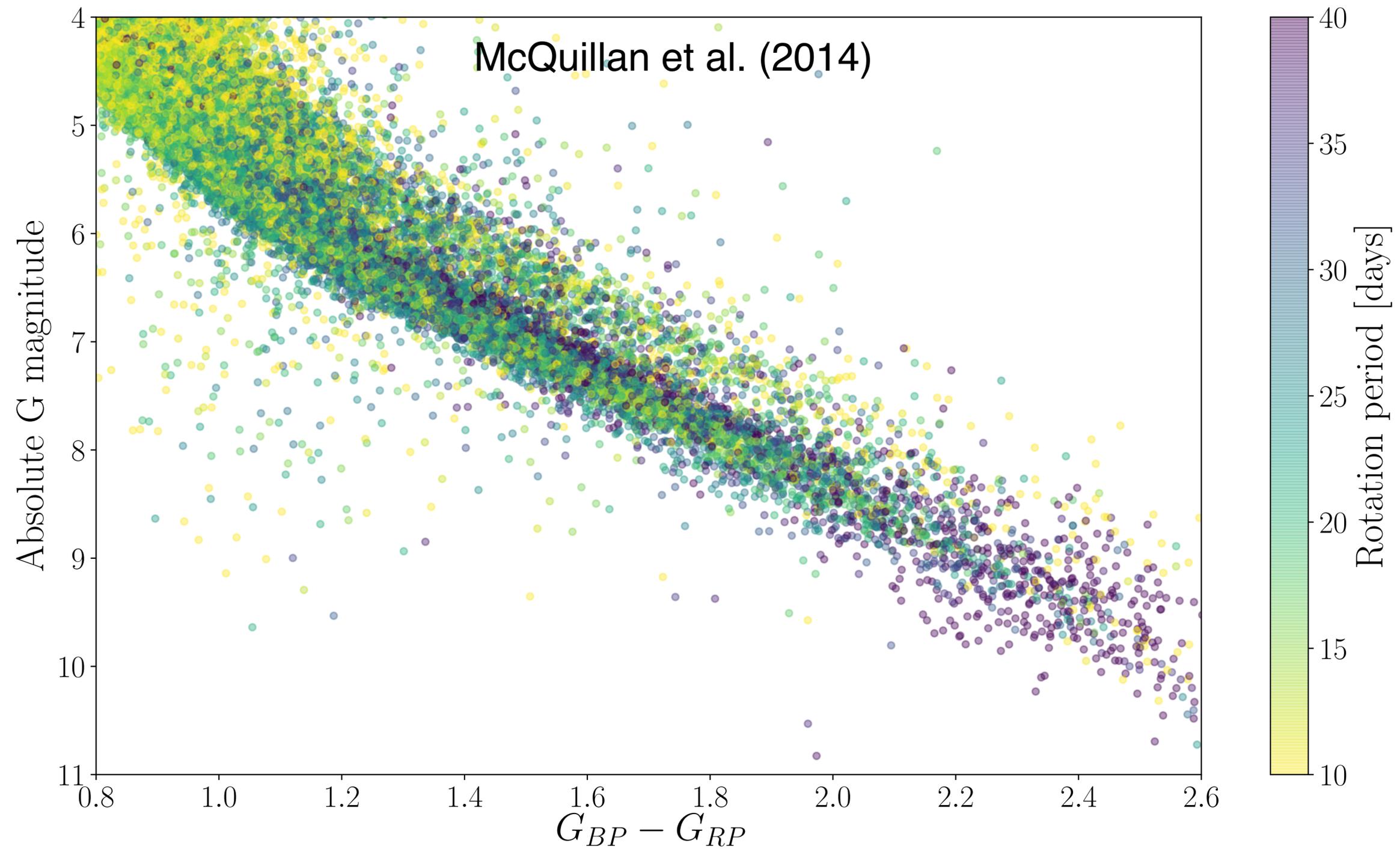
The screenshot shows the documentation for the 'stardate' package. The left sidebar contains a navigation menu with items like 'Installation', 'Running tests', 'Full documentation', 'Quick stardate tutorial: measuring the ages of rotating stars', 'Determining a stellar age', 'Processing and plotting the results.', 'Multiple stars', 'Calculating other physical stellar parameters.', 'Isochrone fitting only', and 'Incorporating asteroseismology'. The main content area has a breadcrumb 'Docs » stardate' and an 'Edit on GitHub' link. The title 'stardate' is followed by the text 'stardate currently only works with python3.' The main text describes 'stardate' as a tool for measuring precise stellar ages, combining isochrone fitting with gyrochronology. It explains that 'stardate' is an extension to 'isochrones' and reverts to 'isochrones' when no rotation period is provided. It provides instructions on how to get started by creating a dictionary of observables and using `stardate.Star.fit()` to infer a Bayesian age. A note mentions that isochrones will be downloaded on first use. The section 'Example usage' is partially visible at the bottom.

Built on [isochrones.py](https://github.com/morton2015/isochrones.py) (Morton, 2018)

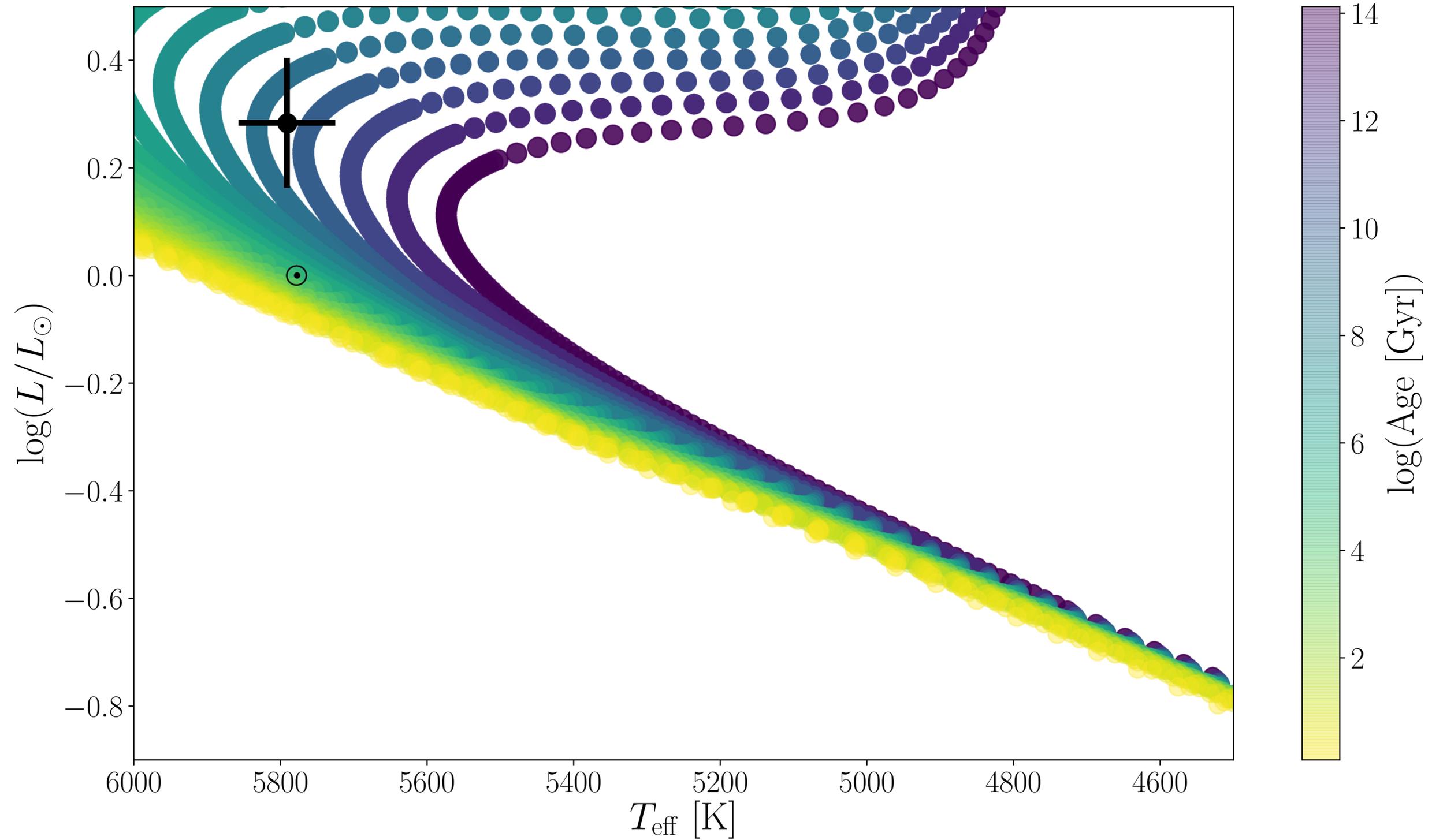
# Stars spin more slowly over time



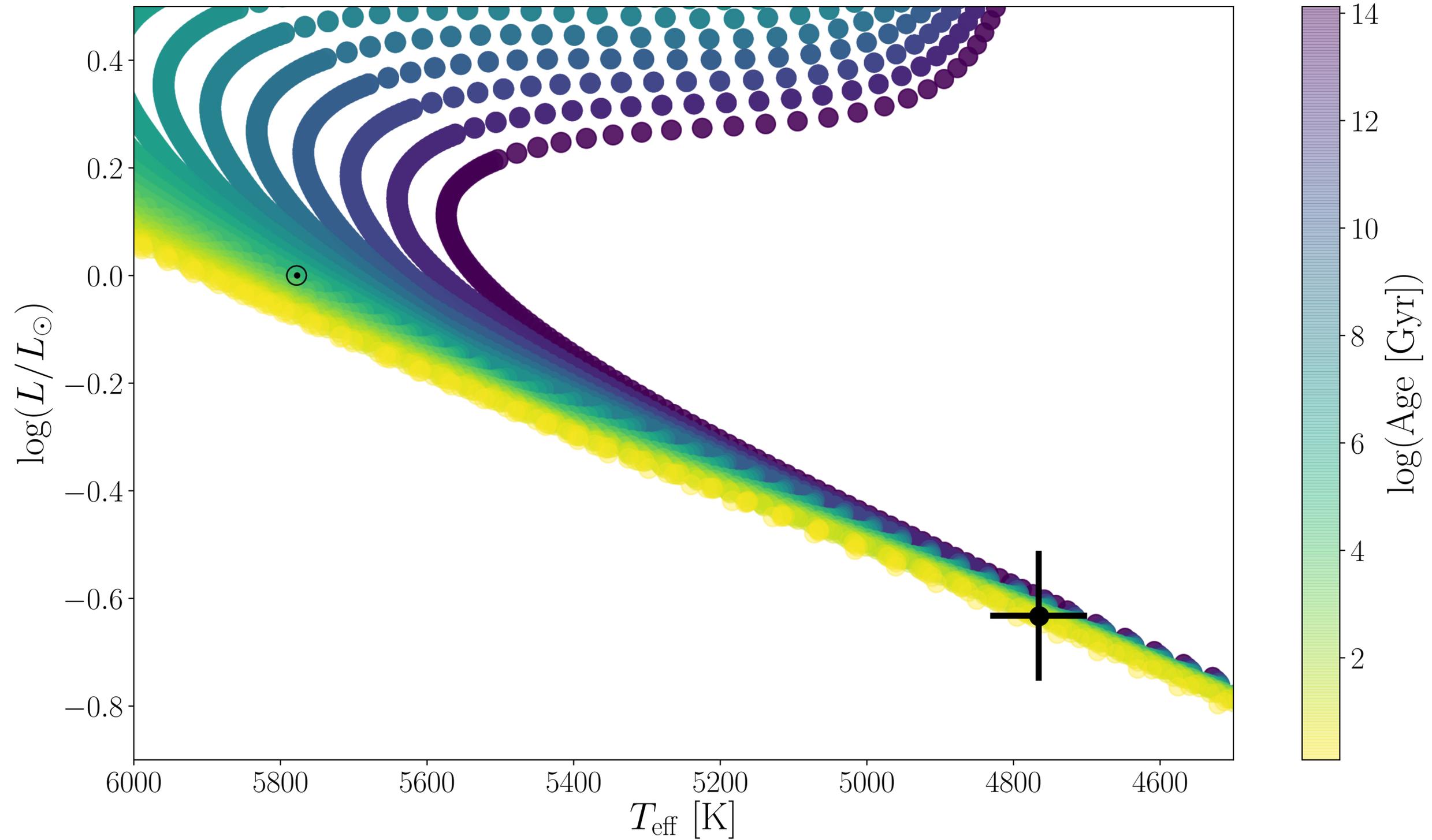
# Stars spin more slowly over time



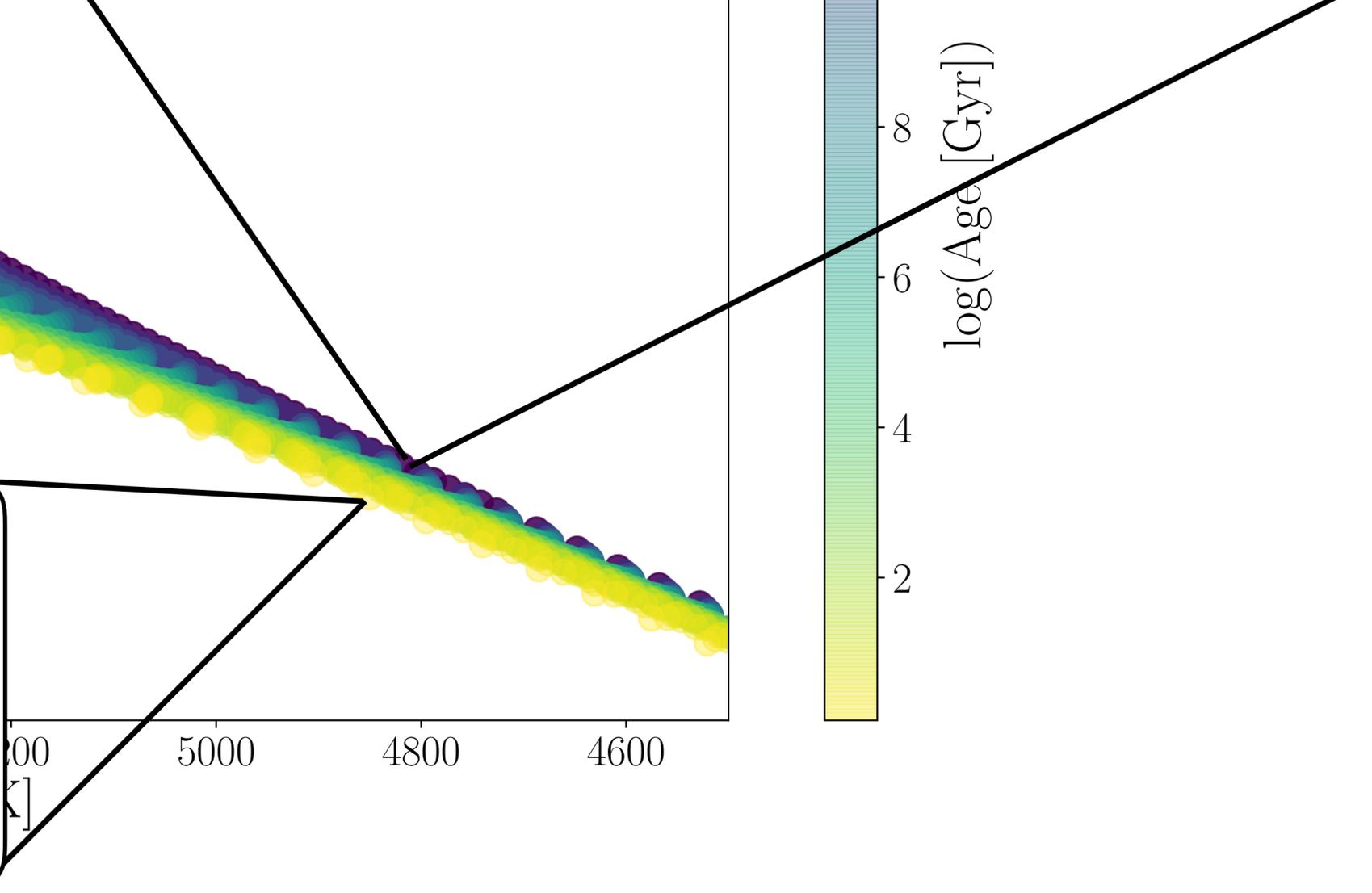
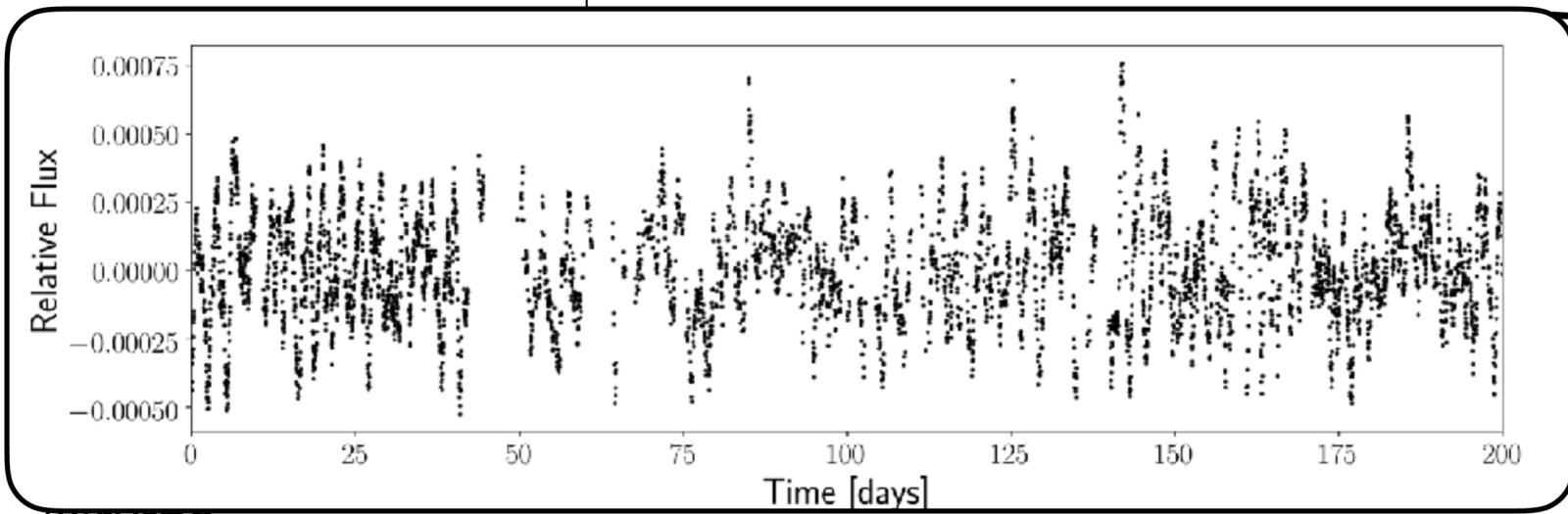
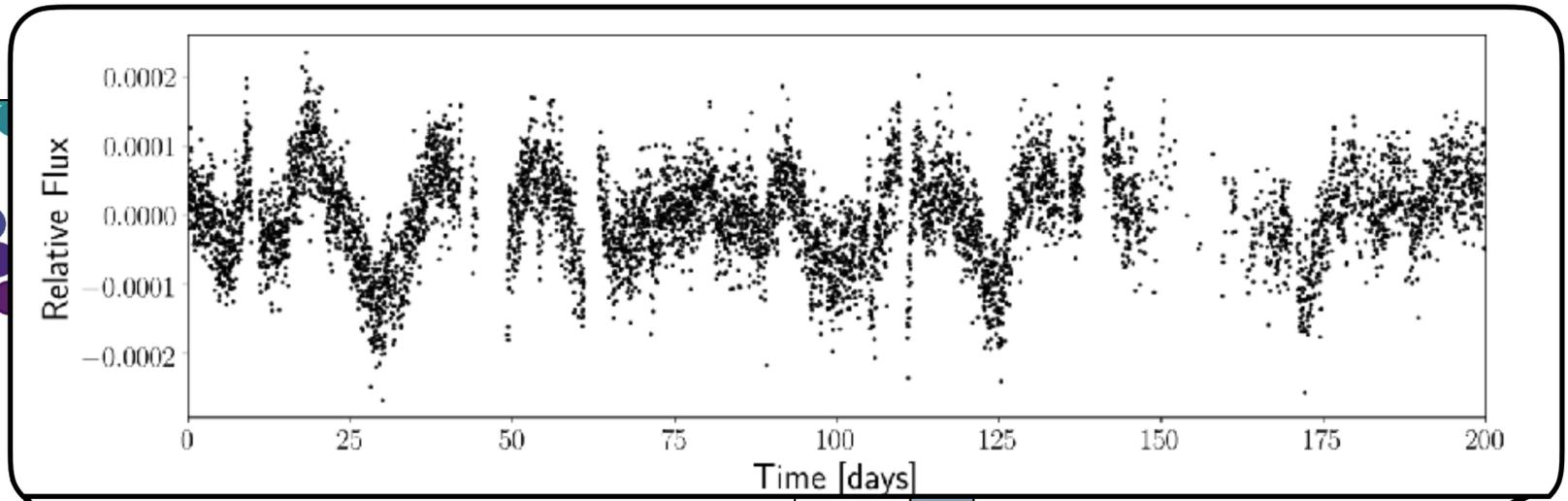
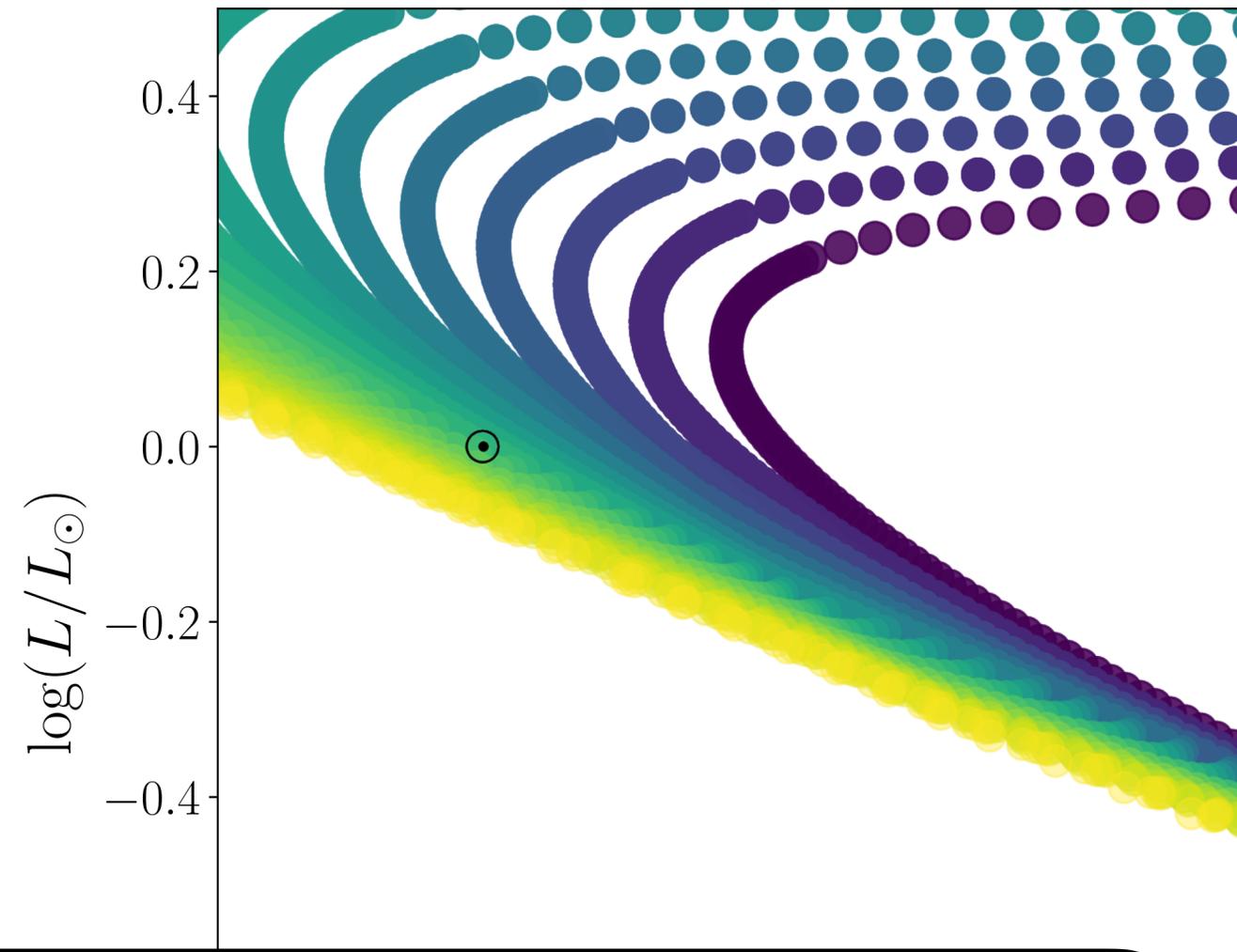
We use MIST isochrones (Dotter + 2016, Choi + 2016)



We use MIST isochrones (Dotter + 2016, Choi + 2016)

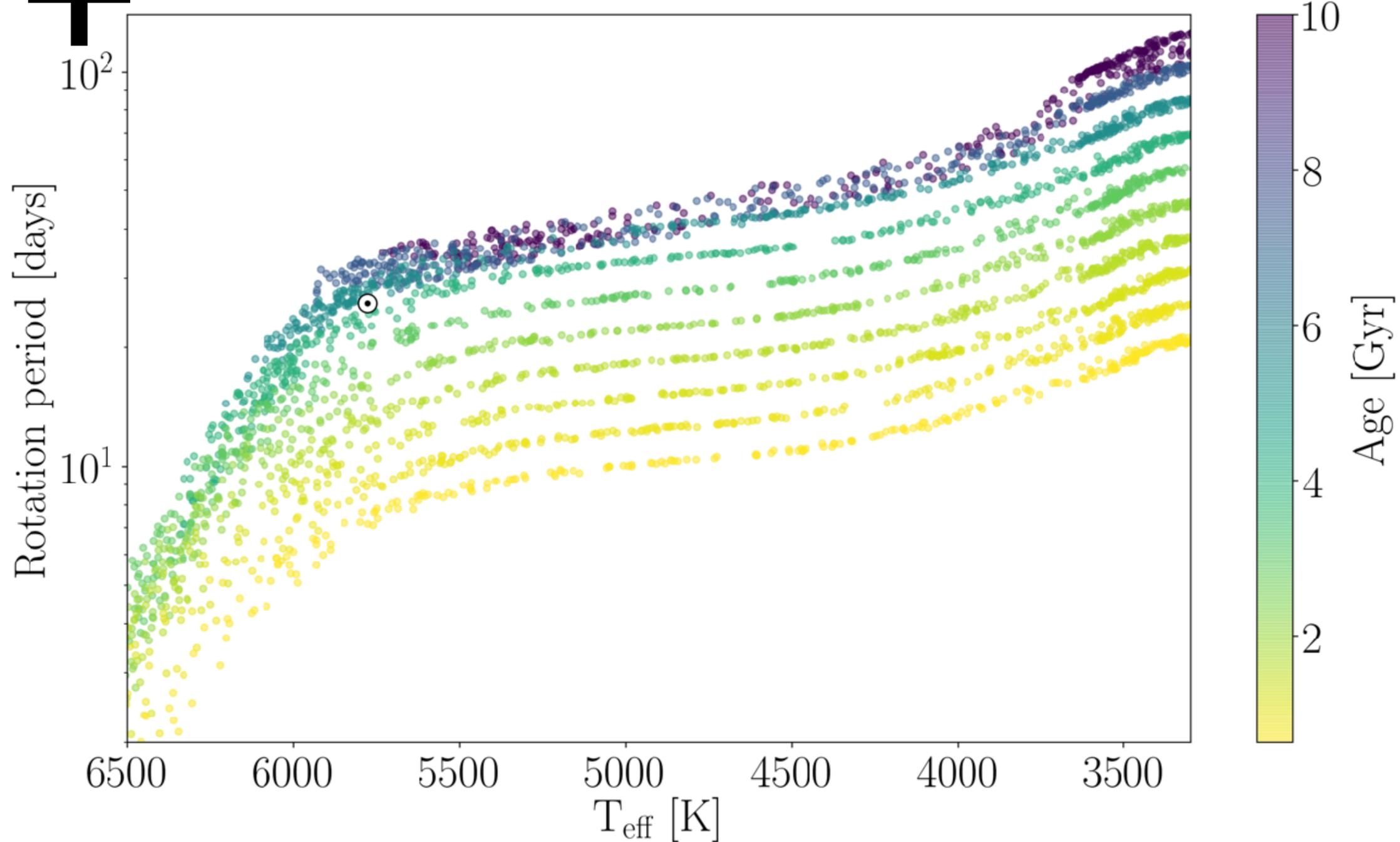
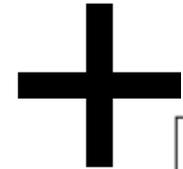
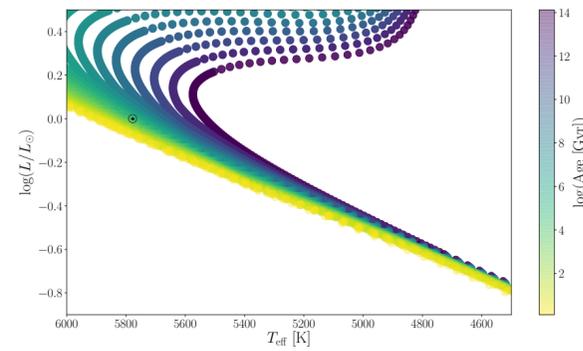


We use MIST isochrones (Dotter + 2016, Choi + 2016)



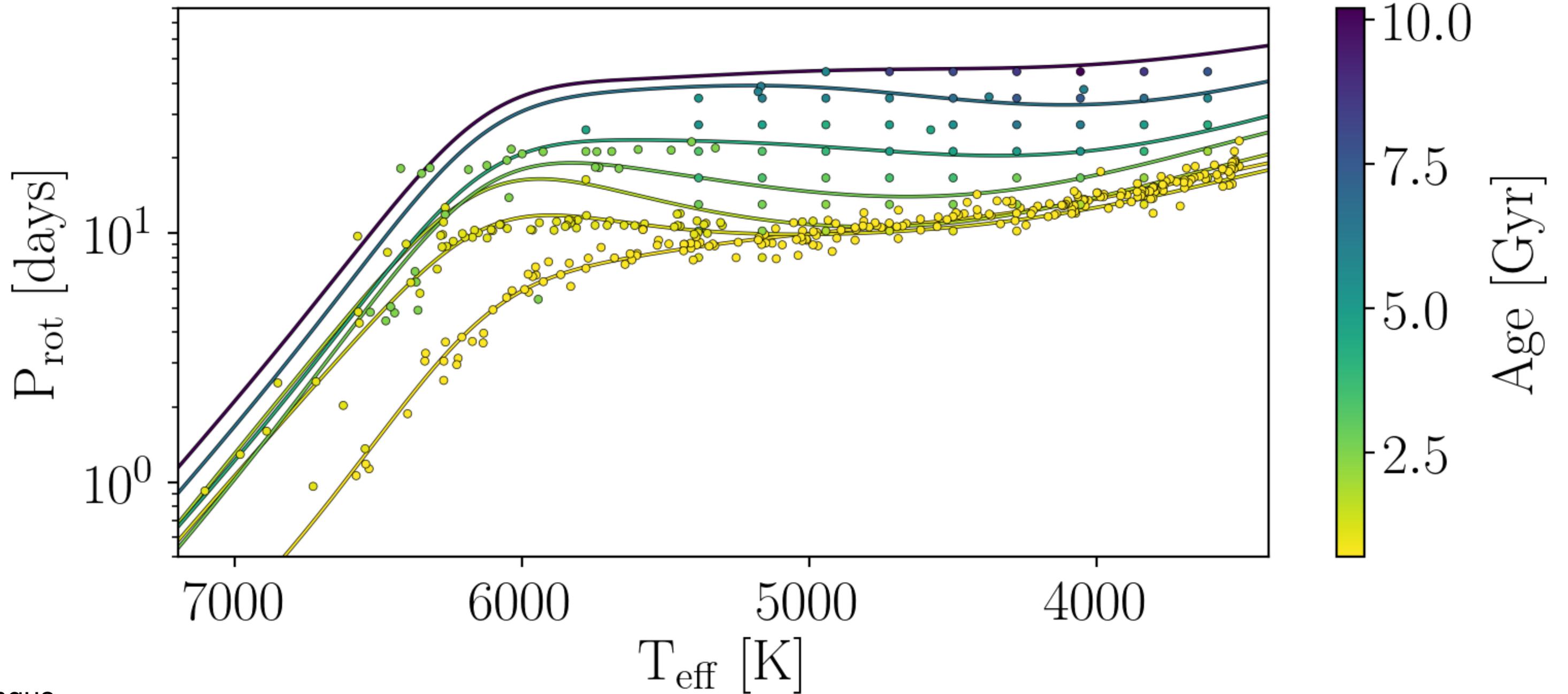
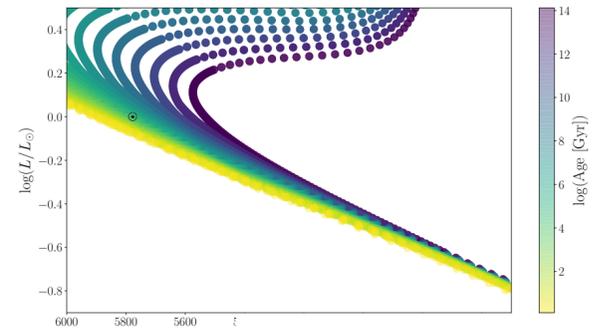
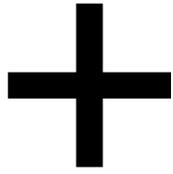
We use MIST isochrones (Dotter + 2016, Choi + 2016)

And a very simple gyrochronology model (Angus + 2019b)



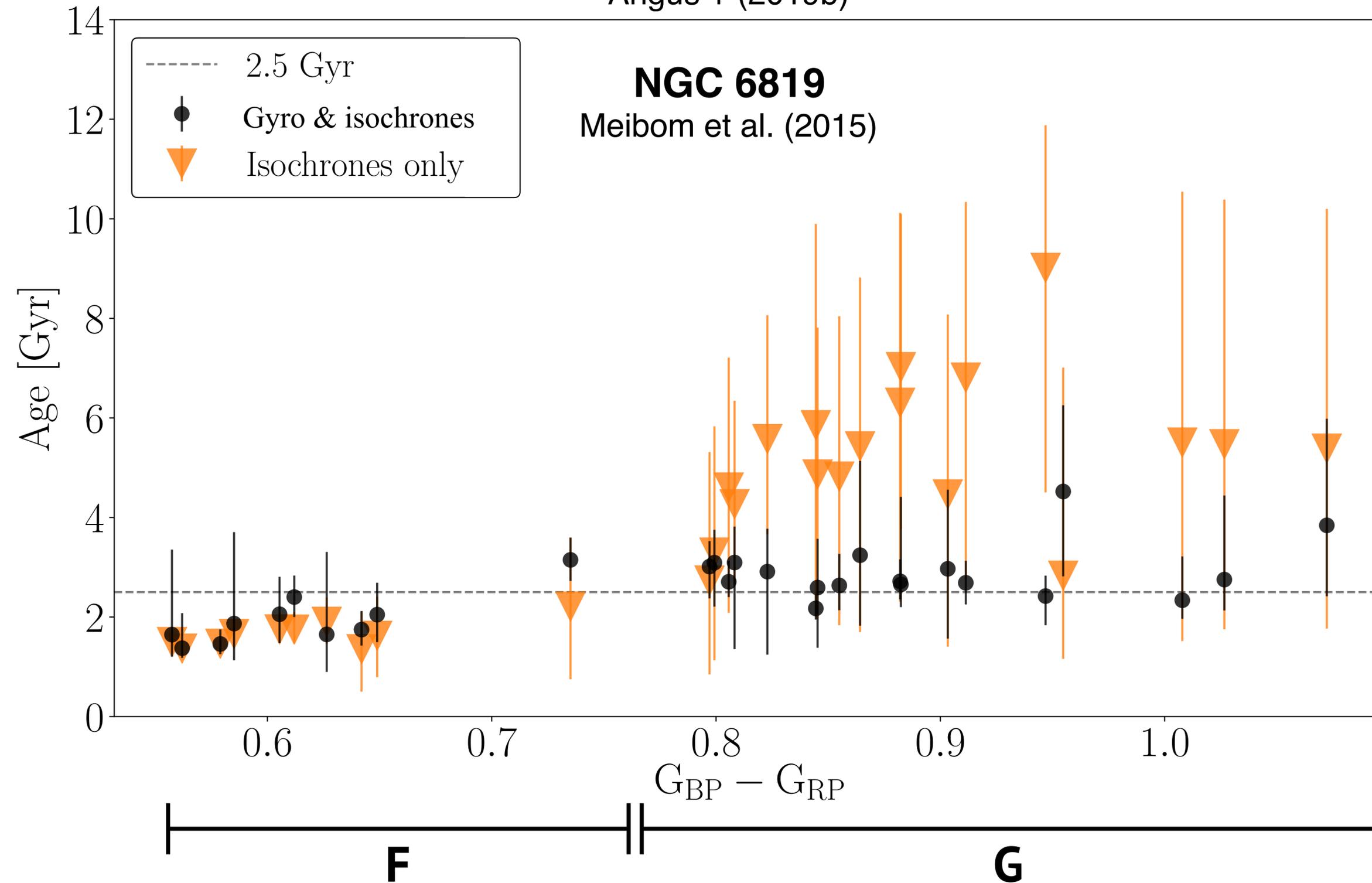
We use MIST isochrones (Dotter + 2016, Choi + 2016)

To be replaced by GP-based gyrochronology model



# Combining methods can be useful

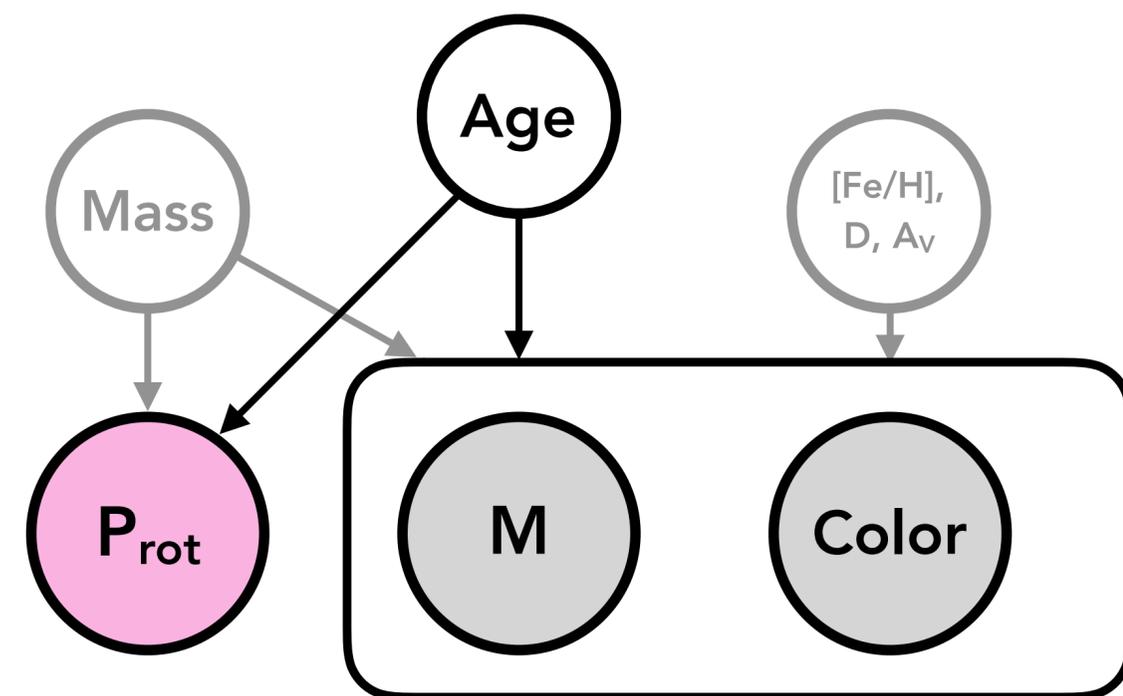
Angus + (2019b)



# Combining methods in stardate

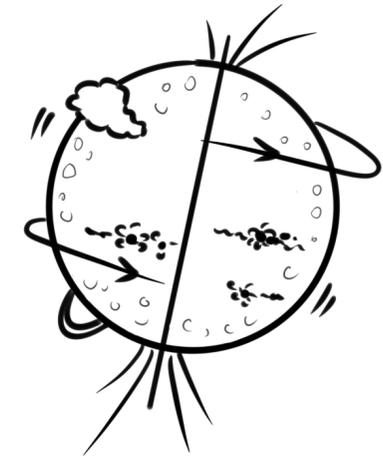
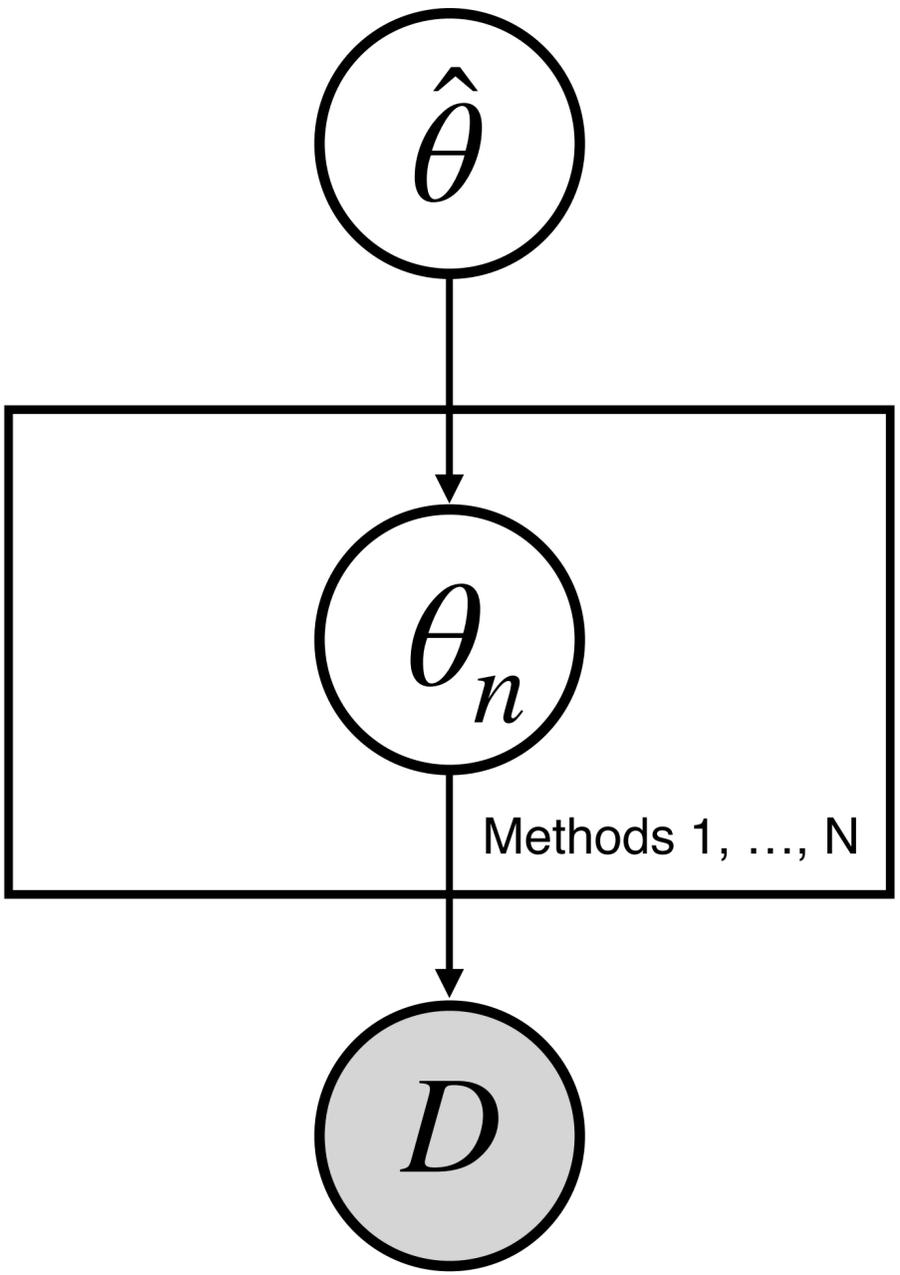
$$p(\theta | D) \propto p(\theta) p(D_{\text{iso}} | \theta) p(P_{\text{rot}} | \theta)$$

This is like augmenting MIST models with a rotation dimension



$\theta = \text{Mass, Age, [Fe/H], etc}$

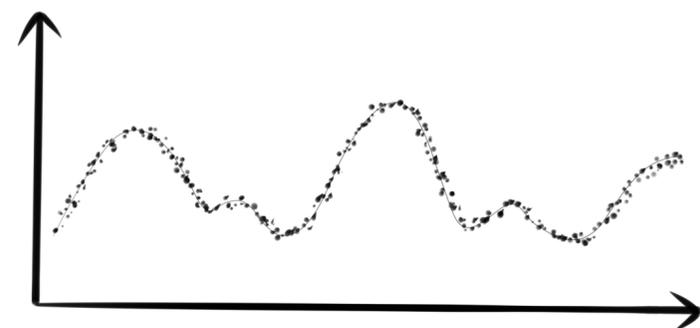
# Combining methods in *general*: a hierarchical problem



True star



Observed star  
(noisy)



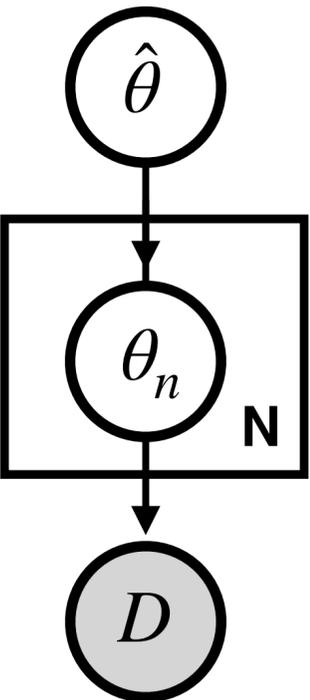
Data

$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

We want:

$$p(\hat{\theta} | D)$$



$\theta$  = Mass, Age, [Fe/H], etc

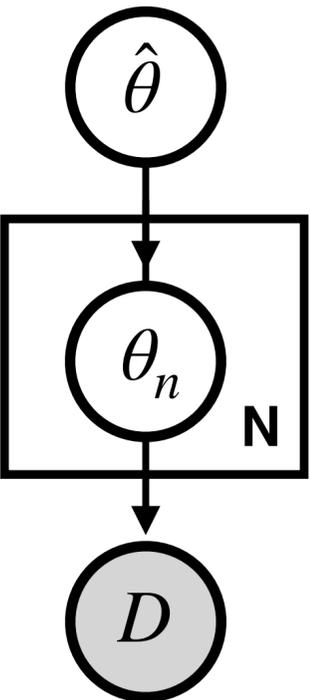
# Combining methods

We want:

$$p(\hat{\theta} | D)$$

We have:

$$p(\hat{\theta}) p(\{\theta_n\} | \hat{\theta}) p(D | \{\theta_n\})$$



$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

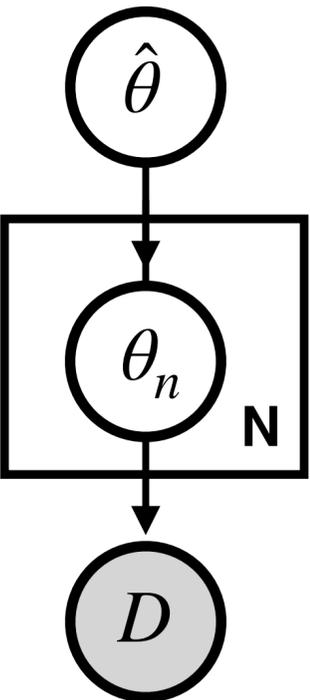
We want:

$$p(\hat{\theta} | D)$$

We have:

$$p(\hat{\theta}) p(\{\theta_n\} | \hat{\theta}) p(D | \{\theta_n\})$$

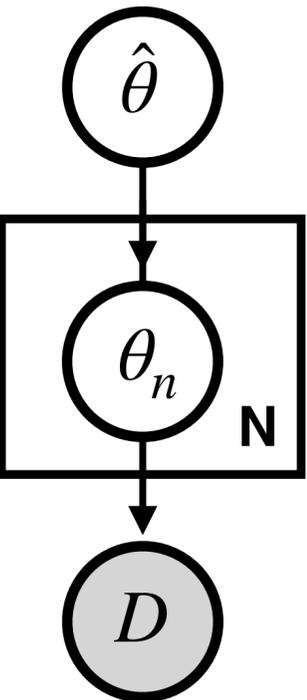
Marginalise over individual parameter measurements



$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

$$p(\hat{\theta} | D) \propto \int p(\hat{\theta}) p(\{\theta_n\} | \hat{\theta}) p(D | \{\theta_n\}) d\{\theta_n\}$$

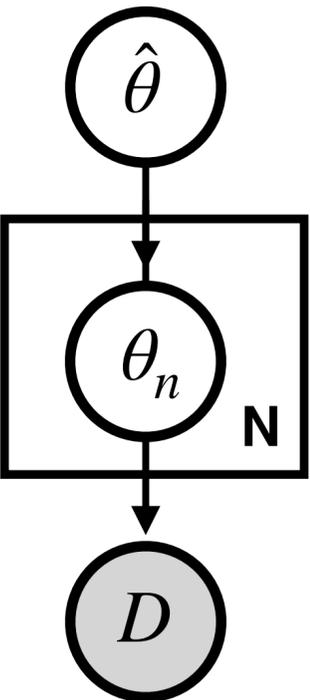


$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \int p(\{\theta_n\} | \hat{\theta}) p(D | \{\theta_n\}) d\{\theta_n\}$$

Prior comes out of the integral

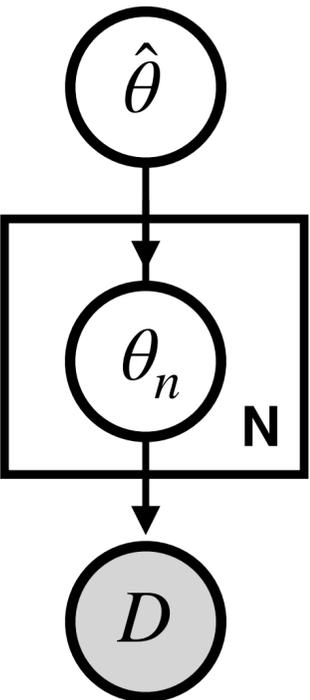


$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \prod_{n=1}^N \int p(\theta_n | \hat{\theta}) p(D | \theta_n) d\theta_n$$

Assumption of independence

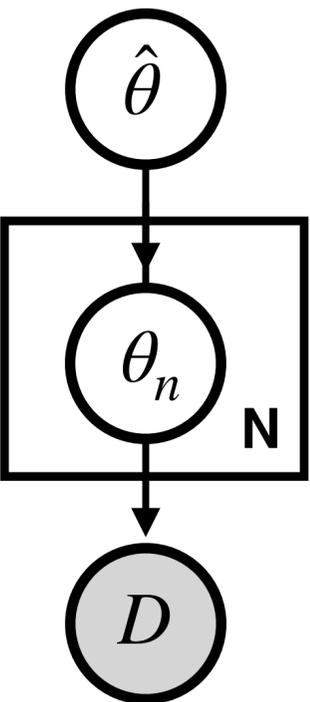
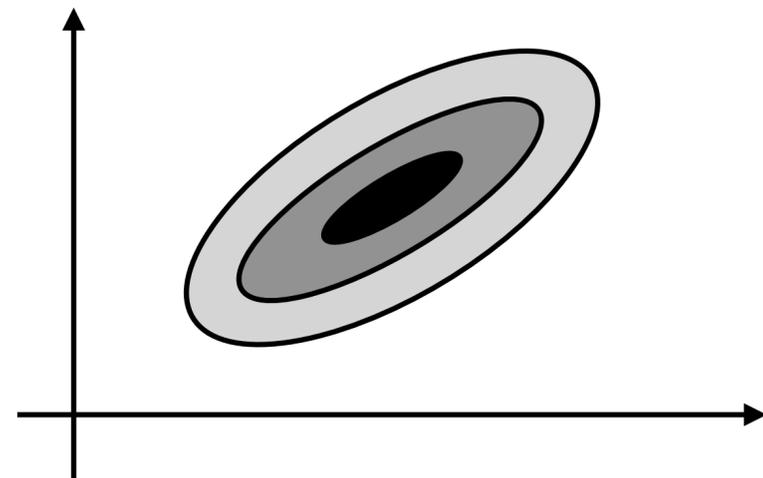


$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \prod_{n=1}^N \int p(\theta_n | \hat{\theta}) \frac{p(D | \theta_n)}{d\theta_n}$$

The likelihood of the data,  
given the measured parameters

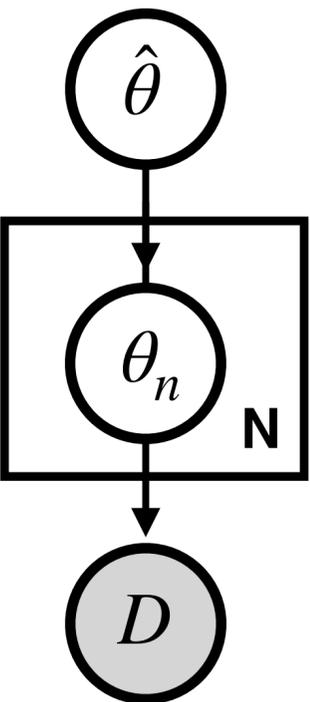
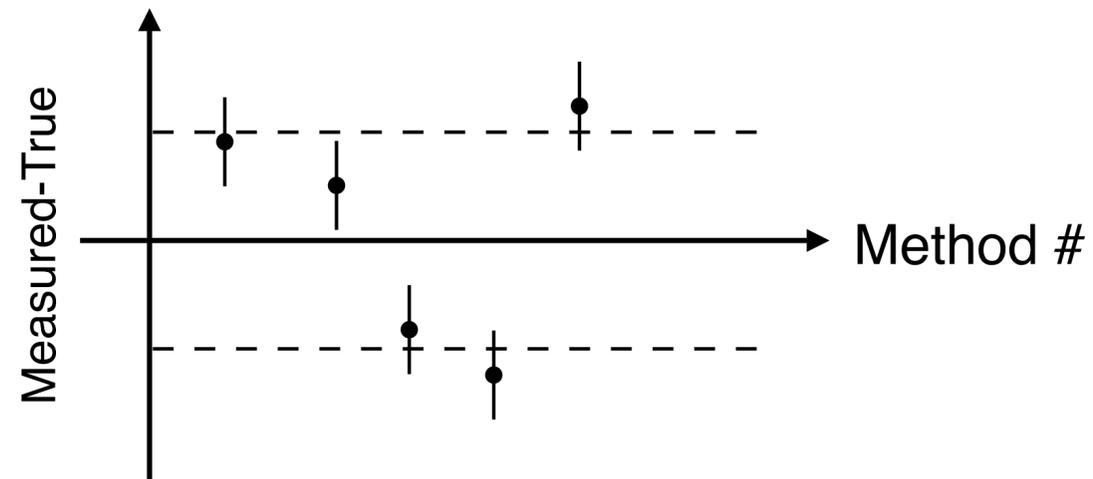


$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \prod_{n=1}^N \int \frac{p(\theta_n | \hat{\theta}) p(D | \theta_n)}{p(\theta_n | \hat{\theta})} d\theta_n$$

The probability of the measured parameters, given the true parameters

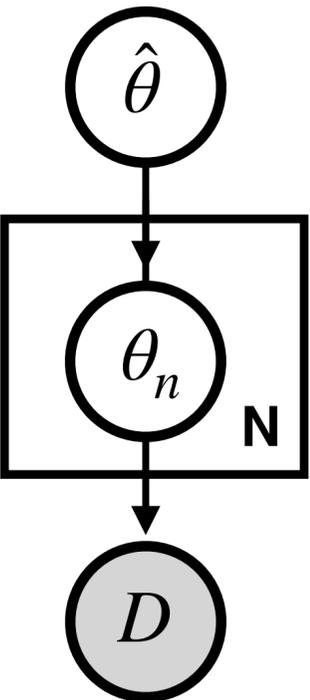


$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods: a hierarchical problem

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \prod_{n=1}^N \int p(\theta_n | \hat{\theta}) p(D | \theta_n) d\theta_n$$

Marginalise over measured parameters

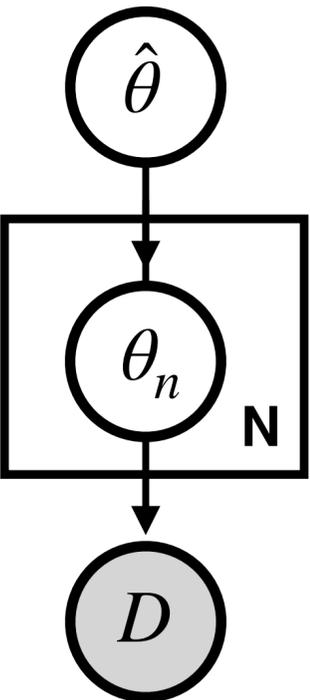


$\theta$  = Mass, Age, [Fe/H], etc

# Combining methods: a hierarchical problem

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \prod_{n=1}^N \int p(\theta_n | \hat{\theta}) p(D | \theta_n) d\theta_n$$

Take product over methods

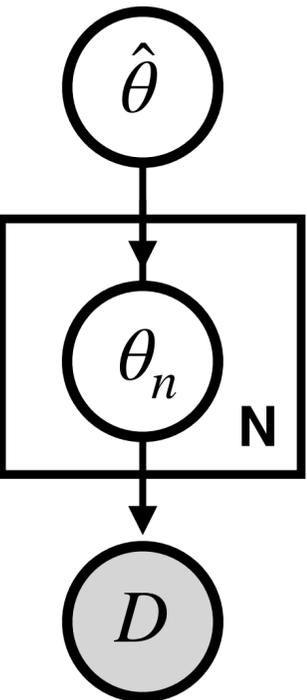


$\theta$  = Mass, Age, [Fe/H], etc

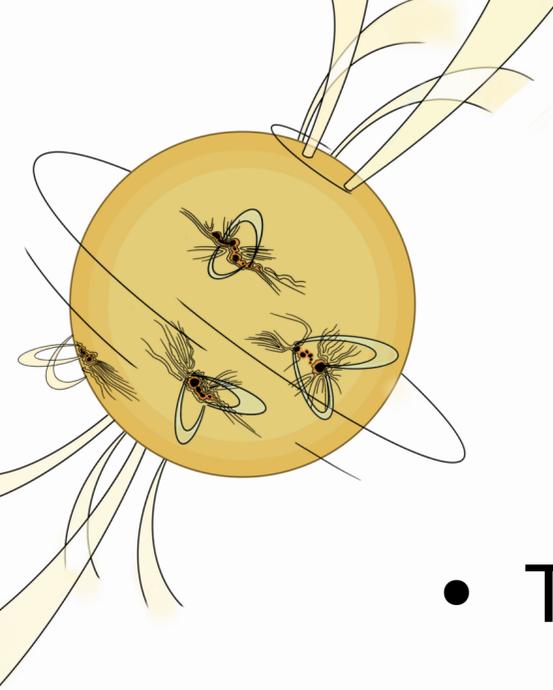
# Combining methods: a hierarchical problem

$$p(\hat{\theta} | D) \propto p(\hat{\theta}) \prod_{n=1}^N \int p(\theta_n | \hat{\theta}) p(D | \theta_n) d\theta_n$$

If you combine models **during** inference,  
this is relatively straightforward



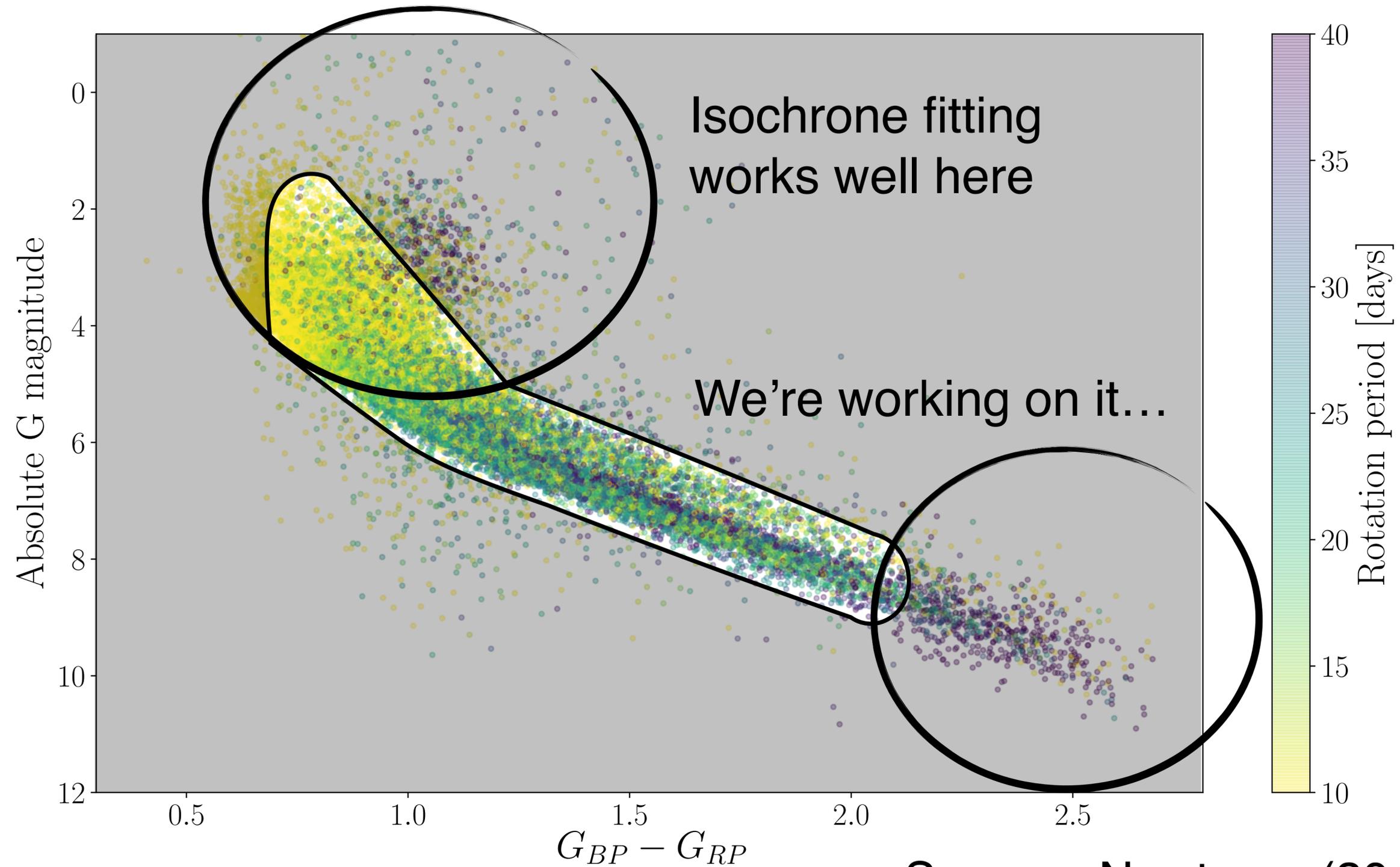
$\theta$  = Mass, Age, [Fe/H], etc



# Summary:

- To combine stellar parameters, marginalise over methods.
- Easiest if likelihood functions are combined directly.
- If not possible, combine approximations to likelihood functions.
- Decide if this is the best approach.
- Publish prior values along with posterior samples.

# stardate **only** applies gyrochronology where calibrated



See e.g. Newton + (2016, 17, 18),  
Kiman + (2019)

# stardate Cross-published in AJ and JOSS)

www.theoj.org/joss-papers/joss.01469/10.21105.joss.01469.pdf

ArXiv: 1908.07528

**stardate.readthedocs.io**  
**github/ruthangus/stardate**

Docs » stardate [Edit on GitHub](#)

## stardate

*stardate* currently only works with python3.

*stardate* is a tool for measuring precise stellar ages. It combines Isochrone fitting with gyrochronology (rotation-based age inference) to increase the precision of stellar ages on the main sequence. The best possible ages provided by *stardate* will be for stars with rotation periods, although ages can be predicted for stars without rotation periods too. If you don't have rotation periods for any of your stars, you might consider using [isochrones.py](#) as *stardate* is simply an extension to *isochrones* that incorporates gyrochronology. *stardate* reverts back to *isochrones* when no rotation period is provided.

In order to get started you can create a dictionary containing the observables you have for your star. These could be atmospheric parameters (like those shown in the example below), or just photometric colors, like those from *2MASS*, *SDSS* or *Gala*. If you have a parallax, asteroseismic parameters, or an idea of the maximum V-band extinction you should throw those in too. Set up the star object and `stardate.Star.fit()` will run Markov Chain Monte Carlo (using *emcee*) in order to infer a Bayesian age for your star.

Note – if you are running the example below for the first time, the isochrones will be downloaded and this could take a while. This will only happen once though!

### Example usage

# stardate Cross-published in AJ and JOSS)

[www.theoj.org/joss-papers/joss.01469/10.21105.joss.01469.pdf](http://www.theoj.org/joss-papers/joss.01469/10.21105.joss.01469.pdf)

ArXiv: 1908.07528

[stardate.readthedocs.io](http://stardate.readthedocs.io)  
[github/ruthangus/stardate](https://github.com/ruthangus/stardate)

Calculating other physical stellar parameters.

Isochrone fitting only

Incorporating asteroseismology

In order to get started you can create a dictionary containing the observables you have for your star. These could be atmospheric parameters (like those shown in the example below), or just photometric colors, like those from *2MASS*, *SDSS* or *Gala*. If you have a parallax, asteroseismic parameters, or an idea of the maximum V-band extinction you should throw those in too. Set up the star object and `stardate.Star.fit()` will run Markov Chain Monte Carlo (using *emcee*) in order to infer a Bayesian age for your star.

Note – if you are running the example below for the first time, the isochrones will be downloaded and this could take a while. This will only happen once though!

## Example usage

```
import stardate as sd

# Create a dictionary of observables
iso_params = {"teff": (4386, 50),      # Teff with uncertainty.
              "logg": (4.66, .05),   # logg with uncertainty.
              "feh": (0.0, .02),     # Metallicity with uncer
              "parallax": (1.48, .1), # Parallax in milliarcsec
              "maxAV": .1}          # Maximum extinction

prot, prot_err = 29, 3

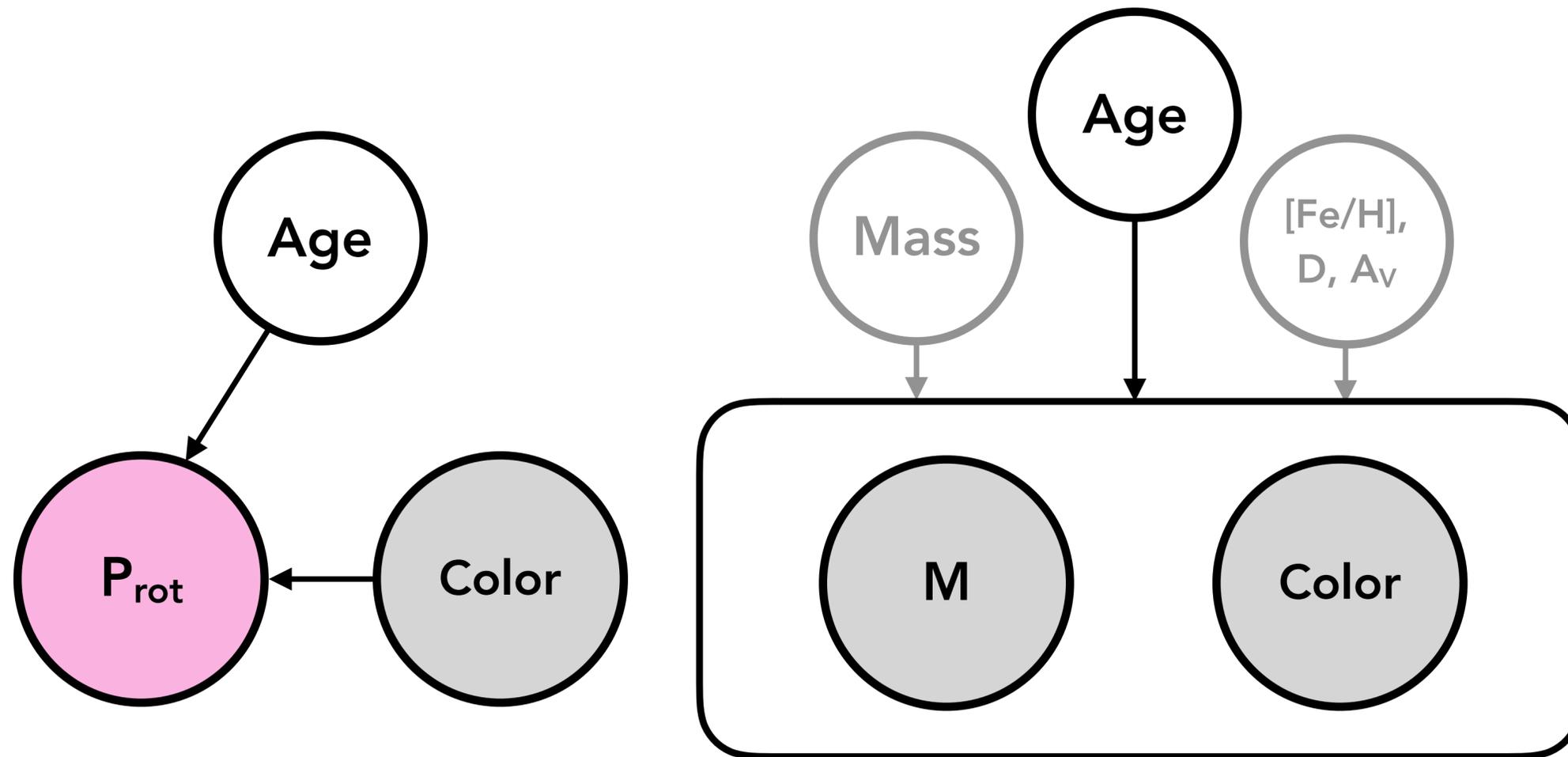
# Set up the star object.
star = sd.Star(iso_params, prot=prot, prot_err=prot_err) # He

# Run the MCMC
star.fit(max_n=1000)
```

# How to **combine** age information

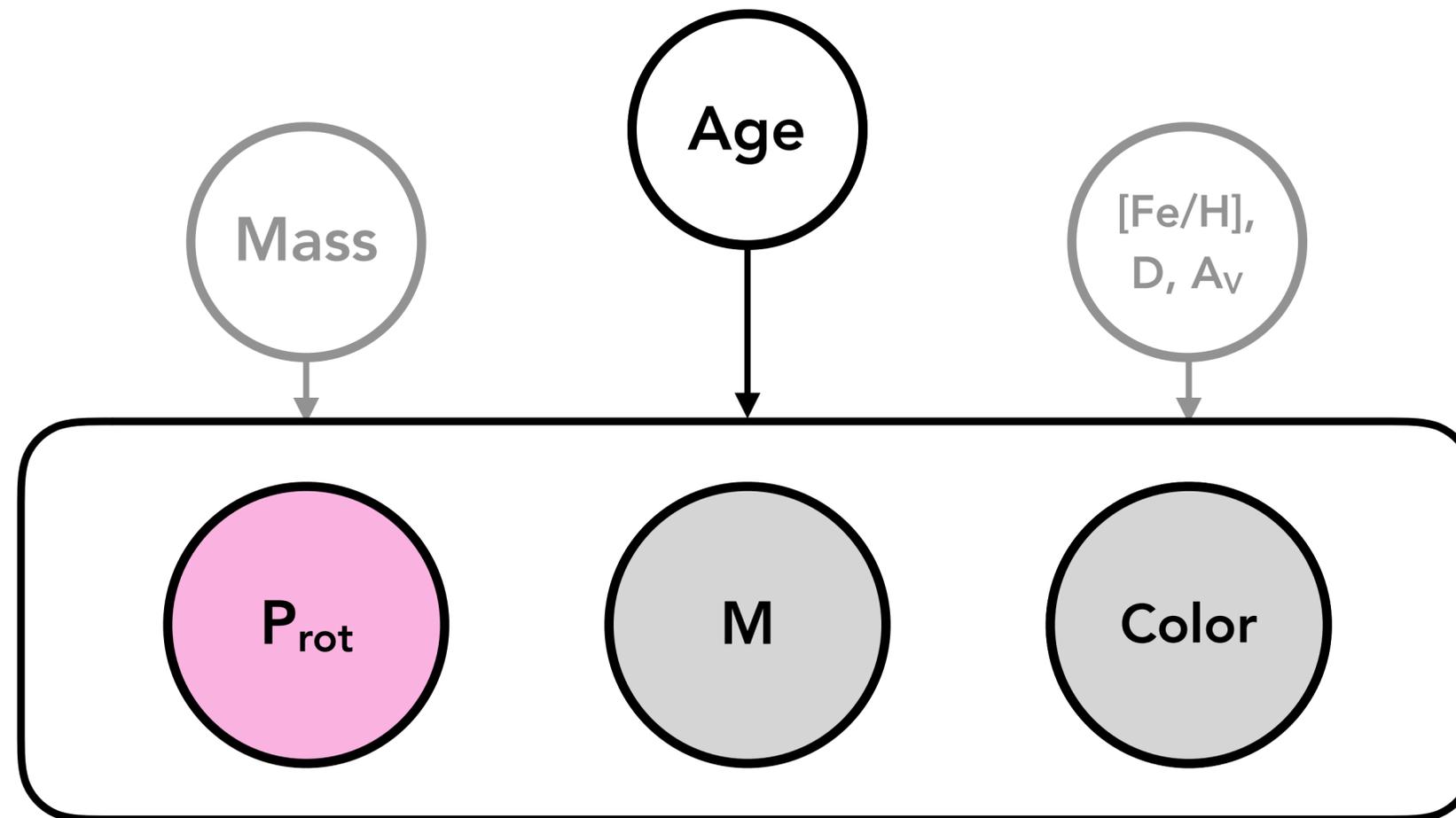
$$\mathcal{L} = p(P_{\text{rot}} | \text{Age, color})$$

$$\mathcal{L} = p(M, \text{color} | \text{Age, Mass, [Fe/H]})$$



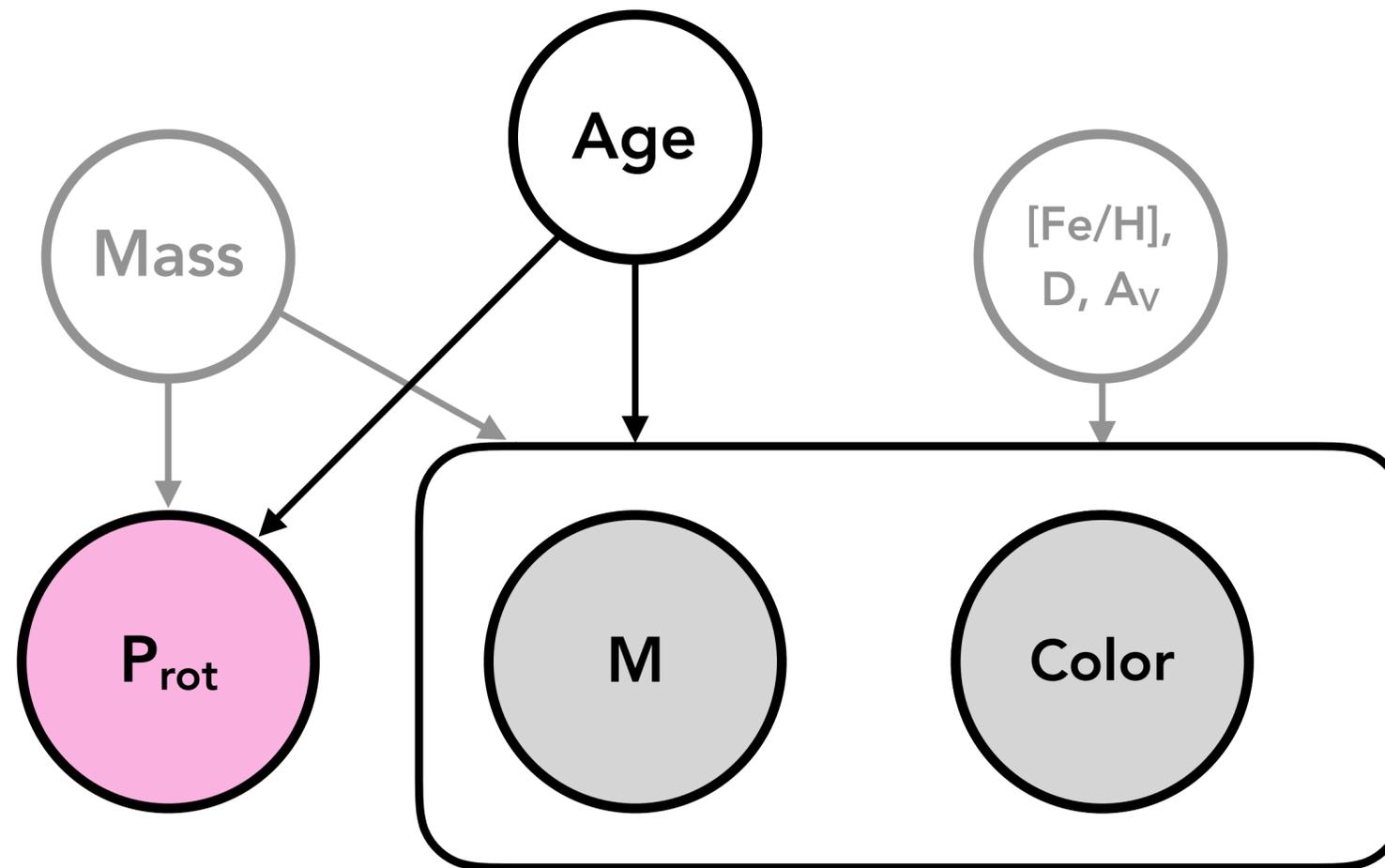
# How to **combine** age information

$$\mathcal{L} = p(M, \text{color}, P_{\text{rot}} | \text{Age}, \text{Mass}, [\text{Fe}/\text{H}])$$



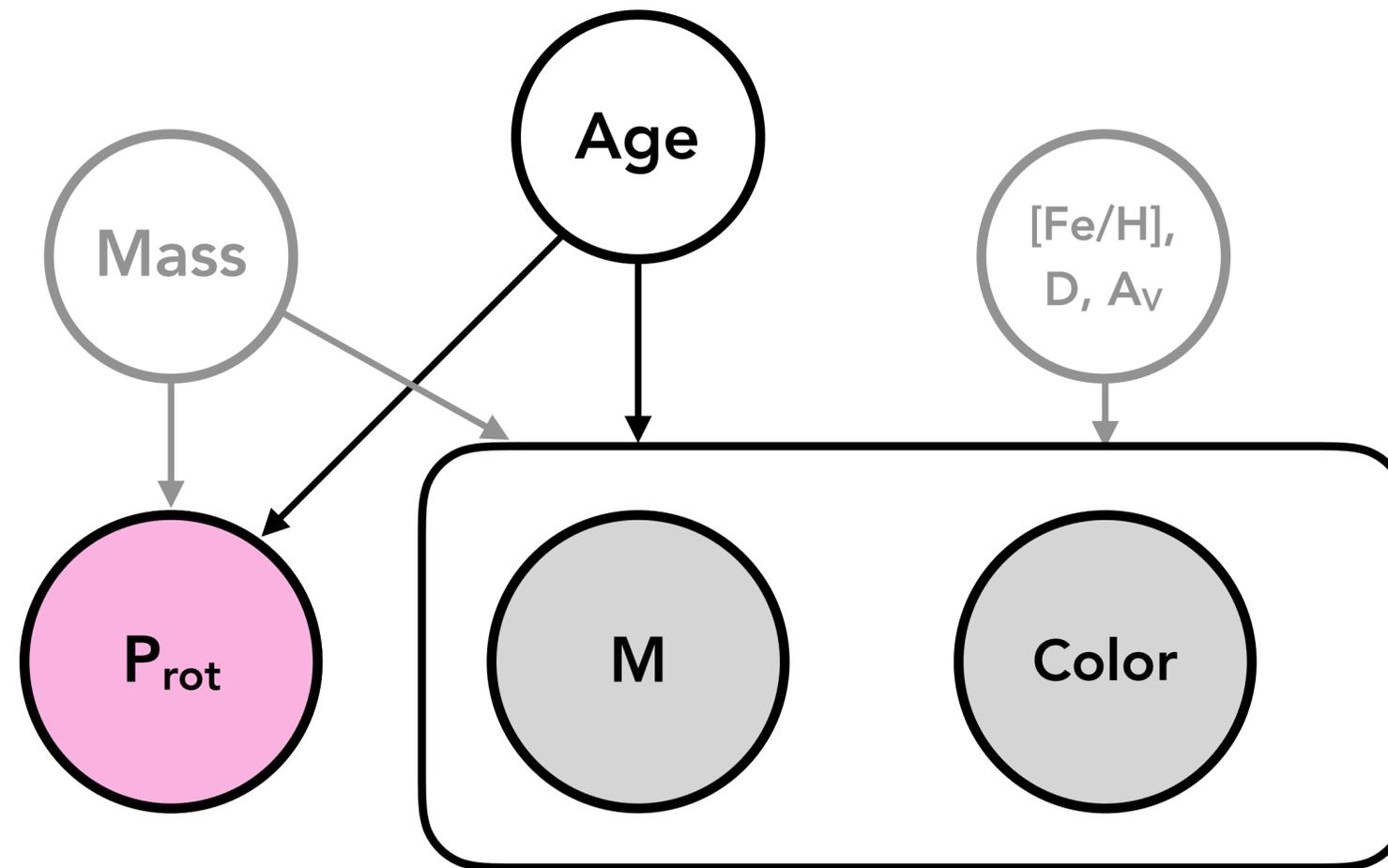
# How to **combine** age information

$$\mathcal{L} = p(M, \text{color} \mid \text{Age}, \text{Mass}, [\text{Fe}/\text{H}]) p(P_{\text{rot}} \mid \text{Age}, \text{Mass})$$



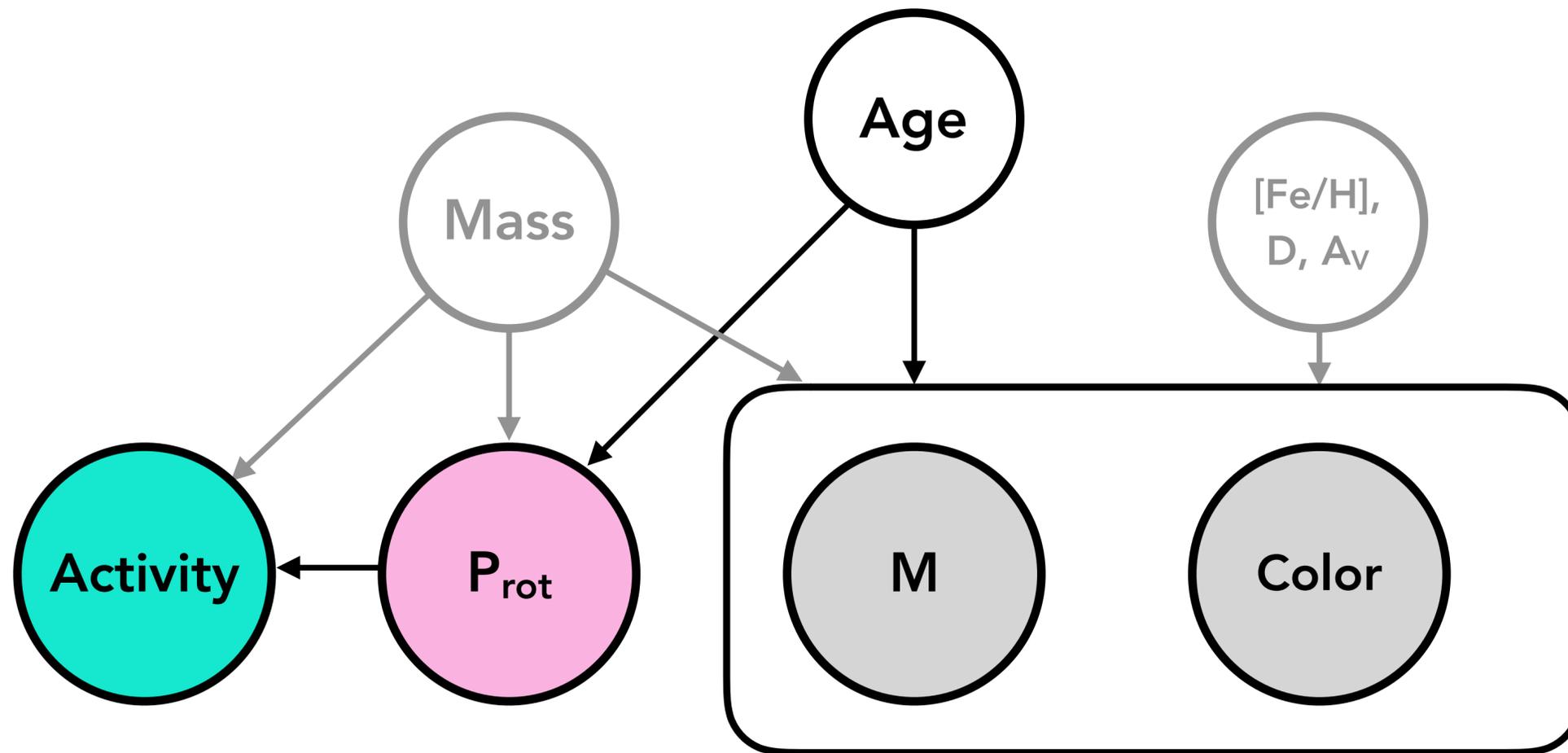
# Adding activity

$$\mathcal{L} = p(M, \text{color} | \text{Age}, \text{Mass}, [\text{Fe}/\text{H}]) p(P_{\text{rot}} | \text{Mass}, \text{Age})$$



# Adding activity

$$\mathcal{L} = p(M, \text{color} | \text{Age}, \text{Mass}, [\text{Fe}/\text{H}]) p(P_{\text{rot}} | \text{Mass}, \text{Age}) p(\text{Activity} | P_{\text{rot}}, \text{Mass})$$



# Adding other age indicators

$$\mathcal{L} = p(M, \text{color} | \text{Age}, \text{Mass}, [\text{Fe}/\text{H}]) p(P_{\text{rot}} | \text{Mass}, \text{Age}) p(\text{Activity} | P_{\text{rot}}, \text{Mass}) \\ \times p(\text{Abundances} | \text{Age}) p(\text{Kinematics} | \text{Age})$$

