NBA Foul Calls and Bayesian Item Response Theory

Posted on April 4, 2017 (../posts/2017-04-04-nba-irt.html)

(**Author's note**: many thanks to Robert ([@atlhawksfanatic](https://twitter.com/atlhawksfanatic) on Twitter) for pointing out some (https://twitter.com/atlhawksfanatic/status/849685639796850689) subtleties (https://twitter.com/atlhawksfanatic/status/849686015753302021) in the data set that I had missed. This post has been revised in line with his feedback. Robert has a very interesting post (http://www.peachtreehoops.com/2017/2/17/14638288/nba-last-two-minute-reports-changing) about how last two minute refereeing has changed over the last three years; I highly recommend you read it.)

I recently found a very interesting data set (https://github.com/polygraph-cool/last-two-minute-report) derived from the NBA's Last Two Minute Report (http://official.nba.com/nba-last-two-minute-reports-archive/) by Russell Goldenberg (http://russellgoldenberg.com/) of The Pudding (https://pudding.cool/). Since 2015, the NBA has released a report reviewing every call and non-call in the final two minutes of every NBA game where the teams were separated by five points or less with two minutes remaining. This data set has extracted each play from the NBA-distributed PDF and augmented it with information from Basketball Reference (https://basketball-reference.com/) to produce a convenient CSV. The Pudding has published two very (https://pudding.cool/2017/02/two-minute-report/) interesting (https://pudding.cool/2017/03/home-court/) visual essays using this data that you should definitely explore.

The NBA is certainly marketed as a star-centric league, so this data set presents a fantastic opportunity to understand the extent to which the players involved in a decision impact whether or not a foul is called. We will also explore other factors related to foul calls.

```
%matplotlib inline
```

```
import datetime
from warnings import filterwarnings
```

```
from matplotlib import pyplot as plt
from matplotlib.ticker import FuncFormatter
import numpy as np
import pandas as pd
import pymc3 as pm
from scipy.special import expit
import seaborn as sns
```

```
blue, green, red, purple, gold, teal = sns.color_palette()

million_dollars_formatter = FuncFormatter(lambda value, _: '${:.1f}M'.format(value / 1e6))
pct_formatter = FuncFormatter(lambda prop, _: "{:.1%}".format(prop))
```

```
filterwarnings('ignore', 'findfont')
```

Loading and preprocessing the data

We begin by loading the data set from GitHub. For reproducibility, we load the data from the most recent commit as of the time this post was published.

```
DATA_URI = 'https://raw.githubusercontent.com/polygraph-cool/last-two-minute-report/1b89b71df060add5538b70d391d7ad82a4c24db2/output/all_games.csv'

raw_df = (pd.read_csv(DATA_URI,
                      usecols=['committing_player', 'disadvantaged_player',
                               'committing_team', 'disadvantaged_team',
                               'seconds_left', 'review_decision', 'date'],
                      parse_dates=['date'])
            .where(lambda df: df.date >= datetime.datetime(2016, 10, 25))
            .dropna(subset=['date'])
            .drop('date', axis=1))
raw_df['review_decision'] = raw_df.review_decision.fillna("INC")
raw_df = (raw_df.dropna()
                .reset_index(drop=True))
```

We restrict our attention to decisions from the 2016-2017 NBA season (https://en.wikipedia.org/wiki/2016%E2%80%9317_NBA_season), for which salary information is readily available (http://www.basketball-reference.com/contracts/players.html) from Basketball Reference (http://www.basketball-reference.com/).

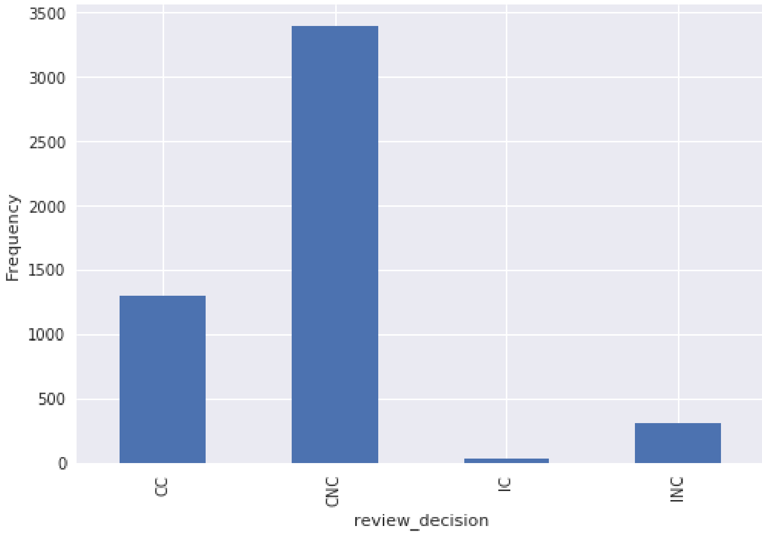
```
raw_df.head()
```

	seconds_left	committing_player	disadvantaged_player	review_decision	disadvantaged_team	committin
0	102.0	Al-Farouq Aminu	George Hill	CNC	UTA	
1	98.0	Boris Diaw	Damian Lillard	CC	POR	
2	64.0	Ed Davis	George Hill	CNC	UTA	
3	62.0	Rudy Gobert	CJ McCollum	INC	POR	
4	27.1	CJ McCollum	Rodney Hood	CC	UTA	

We have only loaded some of the data set's columns; see the original CSV header for the rest.

The response variable in our analysis is derived from `review_decision` , which contains information about whether the incident was a call or non-call and whether, upon post-game review, the NBA deemed the (non-)call correct or incorrect. Below we show the frequencies of each type of `review_decision` .

```
ax = (raw_df.groupby('review_decision')
            .size()
            .plot(kind='bar'))

ax.set_ylabel("Frequency");
```



The possible values of `review_decision` are

- `CC` for correct call,
- `CNC` for correct non-call,
- `IC` for incorrect call, and
- `INC` for incorrect non-call.

While `review_decision` decision provides information about both whether or not a foul was called and whether or not a foul was actually committed, this analysis will focus only on whether or not a foul was called. Including whether or not a foul was actually committed in this analysis introduces some subtleties that are best left to a future post.

In this dataset, the “committing” player is the one that a foul would be called against, if a foul was called on the play, and the other player is “disadvantaged.”

We now encode the data. Since the committing player on one play may be the disadvantaged player on another play, we `melt` (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.melt.html) the raw data frame to have one row per player-play combination so that we can encode the players in a way that is consistent across columns.

```
PLAYER_MAP = {
    "Jose Juan Barea": "JJ Barea",
    "Nene Hilario": "Nene",
    "Tim Hardaway": "Tim Hardaway Jr",
    "James Ennis": "James Ennis III",
    "Kelly Oubre": "Kelly Oubre Jr",
    "Taurean Waller-Prince": "Taurean Prince",
    "Glenn Robinson": "Glenn Robinson III",
    "Otto Porter": "Otto Porter Jr"
}

TEAM_MAP = {
    "NKY": "NYK",
    "COS": "BOS",
    "SAT": "SAS"
}
```

```
long_df = (pd.melt(
                (raw_df.reset_index(drop=True)
                       .rename_axis('play_id')
                       .reset_index()),
                id_vars=['play_id', 'review_decision',
                         'committing_team', 'disadvantaged_team',
                         'seconds_left'],
                value_vars=['committing_player', 'disadvantaged_player'],
                var_name='player', value_name='player_name_')
             # fix inconsistent player names
             .assign(player_name=lambda df: (df.player_name_
                                               .str.replace('\.', '')
                                               .apply(lambda name: PLAYER_MAP.get(name, nam
e))))
             .assign(team_=lambda df: (df.committing_team
                                         .where(df.player == 'committing_player',
                                                df.disadvantaged_team)))
             # fix typos in team names
             .assign(team=lambda df: df.team_.apply(lambda team: TEAM_MAP.get(team, team)))
             .drop(['committing_team', 'disadvantaged_team', 'team_'], axis=1))

long_df['player_id'], player_map = long_df.player_name.factorize()
```

```
long_df.head()
```

	play_id	review_decision	seconds_left	player	player_name_	player_name	team	player_
0	0	CNC	102.0	committing_player	Al-Farouq Aminu	Al-Farouq Aminu	POR	
1	1	CC	98.0	committing_player	Boris Diaw	Boris Diaw	UTA	
2	2	CNC	64.0	committing_player	Ed Davis	Ed Davis	POR	
3	3	INC	62.0	committing_player	Rudy Gobert	Rudy Gobert	UTA	
4	4	CC	27.1	committing_player	CJ McCollum	CJ McCollum	POR	

After encoding, we pivot (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.pivot_table.html) back to a wide data frame with one row per play.

```
df = (long_df.pivot_table(index=['play_id', 'review_decision', 'seconds_left'],
                          columns='player', values='player_id')
             .rename(columns={
                 'committing_player': 'committing_id',
                 'disadvantaged_player': 'disadvantaged_id'
             })
             .rename_axis('', axis=1)
             .reset_index()
             .assign(foul_called=lambda df: 1 * (df.review_decision.isin(['CC', 'IC'])))
             .drop(['play_id', 'review_decision'],
                   axis=1))
```

In addition to encoding the players, we have include a column (`foul_called`) that indicates whether or not a foul was called on the play.

```
df.head()
```

	seconds_left	committing_id	disadvantaged_id	foul_called
0	102.0	0	300	0
1	98.0	1	124	1
2	64.0	2	300	0
3	62.0	3	4	0
4	27.1	4	6	1

In order to understand how foul calls vary systematically across players, we will use salary as a proxy for "star power." The salary data we use was downloaded from Basketball Reference (http://www.basketball-reference.com/contracts/players.html).

```
SALARY_URI = 'http://www.austinrochford.com/resources/nba_irt/2016_2017_salaries.csv'

salary_df = (pd.read_csv(SALARY_URI, skiprows=1,
                         usecols=['Player', '2016-17'])
               .assign(player_name=lambda df: (df.Player
                                                 .str.split('\\', expand=True)[0]
                                                 .str.replace('\.', '')
                                                 # fix inconsistent player names
                                                 .apply(lambda name: PLAYER_MAP.get(name, nam
e))),
                       salary=lambda df: (df['2016-17'].str
                                                        .lstrip('$')
                                                        .astype(np.float64)))
               .assign(log_salary=lambda df: np.log10(df.salary))
               .assign(std_log_salary=lambda df: (df.log_salary - df.log_salary.mean()) / df.l
og_salary.std())
               .drop(['Player', '2016-17'], axis=1)
               .groupby('player_name')
               .max()
               .select(lambda name: name in player_map)
               .assign(player_id=lambda df: (np.equal
                                               .outer(player_map, df.index)
                                               .argmax(axis=0)))
               .reset_index()
               .set_index('player_id')
               .sort_index())
```

Since NBA salaries span many orders of magnitude (LeBron James' salary is just shy of $31M while the lowest paid player made just more than $200K) we will use log salaries, standardized to have mean zero and standard deviation one in our model.

```
salary_df.head()
```

	player_name	salary	log_salary	std_log_salary
player_id				
0	Al-Farouq Aminu	7680965.0	6.885416	0.848869
1	Boris Diaw	7000000.0	6.845098	0.797879
2	Ed Davis	6666667.0	6.823909	0.771080

	player_name	salary	log_salary	std_log_salary
player_id				
3	Rudy Gobert	2121288.0	6.326600	0.142129
4	CJ McCollum	3219579.0	6.507799	0.371293

We also produce a dataframe associating players to teams, along with some useful per-player summaries.

```
team_player_map = (long_df.groupby('team')
                          .player_id
                          .apply(pd.Series.drop_duplicates)
                          .reset_index(level=-1, drop=True)
                          .reset_index()
                          .assign(name=lambda df: player_map[df.player_id],
                                  disadvantaged_rate=lambda tpm_df: (df.groupby('disadvantaged
_id')
                                                                       .foul_called
                                                                       .mean()
                                                                       .ix[tpm_df.player_id]
                                                                       .values),
                                  disadvantaged_plays=lambda tpm_df: (df.groupby('disadvantage
d_id')
                                                                        .size()
                                                                        .ix[tpm_df.player_id]
                                                                        .values))
                          .fillna(0))
```

```
team_player_map.head()
```

	team	player_id	disadvantaged_plays	disadvantaged_rate	name
0	ATL	114	8.0	0.000000	Kyle Korver
1	ATL	115	13.0	0.538462	Dwight Howard
2	ATL	116	44.0	0.272727	Paul Millsap
3	ATL	117	60.0	0.283333	Dennis Schroder
4	ATL	181	25.0	0.200000	Kent Bazemore

Modeling

Throughout this post, we will develop a series of models for understanding how foul calls vary across players, starting with a simple beta-Bernoulli model and working our way up to a hierachical item-response theory (https://en.wikipedia.org/wiki/Item_response_theory) regression model.

Before building models, we must introduce a bit of notation. The index i will correspond to a disadvantaged player and the index j corresponds to a committing player. The index k corresponds to a play. With this notation $i(k)$ and $j(k)$ are the index of the disadvantaged and committing player involved in play k, respectively. The binary variable y_k indicates whether or not a foul was called on play k. All of our models use the likelihood

$$y_k \sim \text{Bernoulli}(p_{i(k),j(k)}).$$

Each model differs in its specification of the probability that a foul is called, $p_{i,j}$.

Beta-Bernoulli model

One of the simplest possible models for this data focuses only on the disadvantaged player, so $p_{i,j} = p_i$, and places independent beta priors on each p_i. For simplicity, we begin with uniform priors, $p_i \sim \text{Beta}(1, 1)$.

Even though this model is conjugate, we will use `pymc3` (http://pymc-devs.github.io/pymc3/) to perform inference with it for consistency with subsequent, non-conjugate models.

```
n_players = player_map.size
disadvantaged_id = df.disadvantaged_id.values

foul_called = df.foul_called.values
obs_rate = foul_called.mean()
```

```
with pm.Model() as bb_model:
    p = pm.Beta('p', 1., 1., shape=n_players)

    y = pm.Bernoulli('y_obs', p[disadvantaged_id],
                     observed=foul_called)
```

Throughout this post, we will use the no-U-turn sampler (https://arxiv.org/abs/1111.4246) for inference, tuning the sampler's hyperparameters for the first two thousand samples and subsequently keeping the next two thousand samples for inference.

```
N_TUNE = 2000
N_SAMPLES = 2000

SEED = 506421 # from random.org, for reproducibility
```

We now sample from the beta-Bernoulli model.

```
def sample(model, n_tune, n_samples, seed):
    with model:
        full_trace = pm.sample(n_tune + n_samples, tune=n_tune, random_seed=seed)

    return full_trace[n_tune:]
```

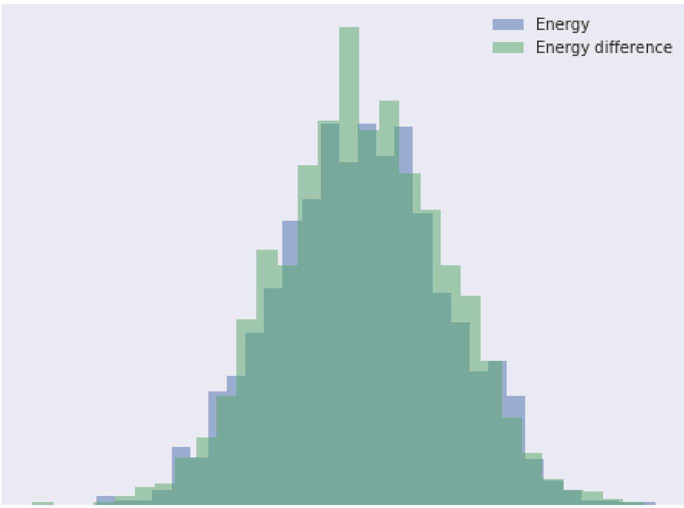
```
bb_trace = sample(bb_model, N_TUNE, N_SAMPLES, SEED)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using advi...
  2%|▏         | 4793/200000 [00:02<01:47, 1818.84it/s]Median ELBO converged.
Finished [100%]: Average ELBO = -3,554.3

100%|██████████| 4000/4000 [01:03<00:00, 63.38it/s]
```

We use energy energy plots to diagnose possible problems with our samples.

```python
def energy_plot(trace):
    energy = trace['energy']
    energy_diff = np.diff(energy)

    fig, ax = plt.subplots(figsize=(8, 6))

    ax.hist(energy - energy.mean(), bins=30,
            lw=0, alpha=0.5,
            label="Energy")
    ax.hist(energy_diff, bins=30,
            lw=0, alpha=0.5,
            label="Energy difference")

    ax.set_xticks([])
    ax.set_yticks([])

    ax.legend()
```

```python
energy_plot(bb_trace)
```



Since the energy and energy difference distributions are quite similar, there is no indication from this plot of sampling issues. For an in-depth treatment of Hamiltonian Monte Carlo algorithms and convergence diagnostics, consult Michael Betancourt (https://betanalpha.github.io/)'s excellent paper *A Conceptual Introduction to Hamiltonian Monte Carlo* (https://arxiv.org/abs/1701.02434).

We will use the widely applicable information criterion (http://www.stat.columbia.edu/~gelman/research/unpublished/loo_stan.pdf) (WAIC) and binned residuals to check and compare our models. WAIC is a Bayesian measure of out-of-sample predictive accuracy based on in-sample data that is quite closely related to [leave-one-out cross-validation] (https://en.wikipedia.org/wiki/Cross-validation_(statistics%29#Leave-one-out_cross-validation). It attempts to improve upon known shortcomings of the widely-used deviance information criterion (https://en.wikipedia.org/wiki/Deviance_information_criterion). (See *Understanding predictive information criteria for Bayesian models* (http://www.stat.columbia.edu/~gelman/research/published/waic_understand3.pdf) for a review and comparison of various information criteria, including DIC and WAIC.) WAIC is easy to calculate with `pymc3`.

```
def get_waic_df(model, trace, name):
    with model:
        waic = pm.waic(trace)

    return pd.DataFrame.from_records([waic], index=[name], columns=waic._fields)
```

```
waic_df = get_waic_df(bb_model, bb_trace, "Beta-Bernoulli")
```

```
/opt/conda/lib/python3.5/site-packages/pymc3/stats.py:145: UserWarning: For one or more sample
s the posterior variance of the
        log predictive densities exceeds 0.4. This could be indication of
        WAIC starting to fail see http://arxiv.org/abs/1507.04544 for details

  """)
```

We see that the WAIC calculation indicates difficulties with the beta-Bernoulii model, which we will soon confirm.

```
waic_df
```

	WAIC	WAIC_se	p_WAIC
Beta-Bernoulli	6021.491064	66.23822	238.488377

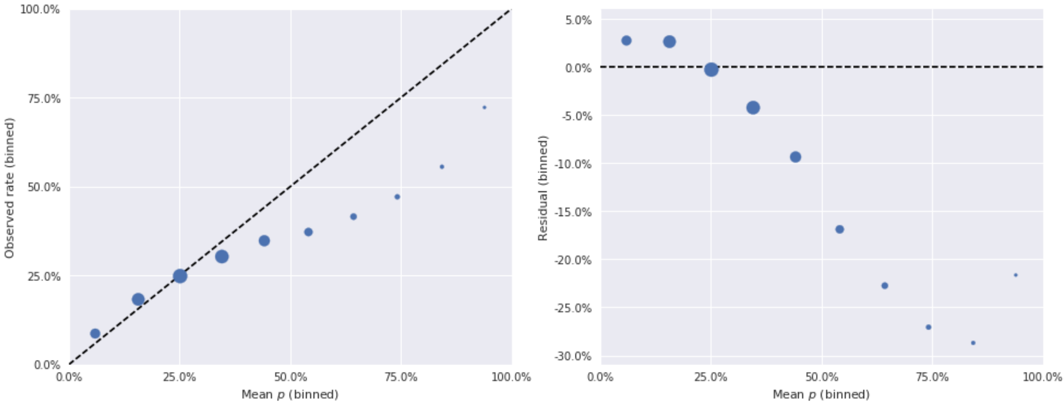
In addition to the WAIC value, we get an estimate of its standard error (`WAIC_se`) and the number of effective parameters in the model (`p_WAIC`). The number of effective parameters is an indication of model complexity.

The second diagnostic tool we use on our models are binned residuals, which show how well-calibrated the model's predicted probabilities are. Intuitively, if our model predicts that an event has a 35% chance of occurring and we can observe many repetitions of that event, we would expect the event to actually occur about 35% of the time. If the observed occurrences of the event differ substantially from the predicted rate, we have reason to doubt the quality of our model. Since we generally can't observe each event many times, we instead group events into bins by their predicted probability and check that the average predicted probability in each bin is close to the rate at which the events in that bin are observed.

The binned predictions and residuals for the beta-Bernoulli model are shown below.

```python
BINS = np.linspace(0, 1, 11)
BINS_3D = BINS[np.newaxis, np.newaxis]

def binned_residuals(y, p):
    p_3d = p[..., np.newaxis]

    in_bin = (BINS_3D[..., :-1] < p_3d) & (p_3d <= BINS_3D[..., 1:])
    bin_counts = in_bin.sum(axis=(0, 1))

    p_mean = (in_bin * p_3d).sum(axis=(0, 1)) / bin_counts
    y_mean = (in_bin * y[np.newaxis, :, np.newaxis]).sum(axis=(0, 1)) / bin_counts

    return y_mean, p_mean, bin_counts

def binned_residual_plot(bin_obs, bin_p, bin_counts):
    fig, (ax, resid_ax) = plt.subplots(ncols=2, sharex=True, figsize=(16, 6))

    ax.scatter(bin_p, bin_obs,
               s=300 * np.sqrt(bin_counts / bin_counts.sum()),
               zorder=5)
    ax.plot([0, 1], [0, 1], '--', c='k')

    ax.set_xlim(0, 1)
    ax.set_xticks(np.linspace(0, 1, 5))
    ax.xaxis.set_major_formatter(pct_formatter)
    ax.set_xlabel("Mean $p$ (binned)")

    ax.set_ylim(0, 1)
    ax.set_yticks(np.linspace(0, 1, 5))
    ax.yaxis.set_major_formatter(pct_formatter)
    ax.set_ylabel("Observed rate (binned)")

    resid_ax.scatter(bin_p, bin_obs - bin_p,
                     s=300 * np.sqrt(bin_counts / bin_counts.sum()),
                     zorder=5)

    resid_ax.hlines(0, 0, 1, 'k', '--')

    resid_ax.set_xlim(0, 1)
    resid_ax.set_xticks(np.linspace(0, 1, 5))
    resid_ax.xaxis.set_major_formatter(pct_formatter)
    resid_ax.set_xlabel("Mean $p$ (binned)")

    resid_ax.yaxis.set_major_formatter(pct_formatter)
    resid_ax.set_ylabel("Residual (binned)")
```

```python
bin_obs, bin_p, bin_counts = binned_residuals(foul_called, bb_trace['p'][:, disadvantaged_id])

binned_residual_plot(bin_obs, bin_p, bin_counts)
```



In these plots, the dashed black lines show how these quantities would be related, for a perfect model. The area of each point is proportional to the number of events whose predicted probability fell in the relevant bin. From these plots, we get further confirmation that our simple beta-Bernoulli model is quite unsatisfactory, as many binned residuals exceed 5% in absolute value.

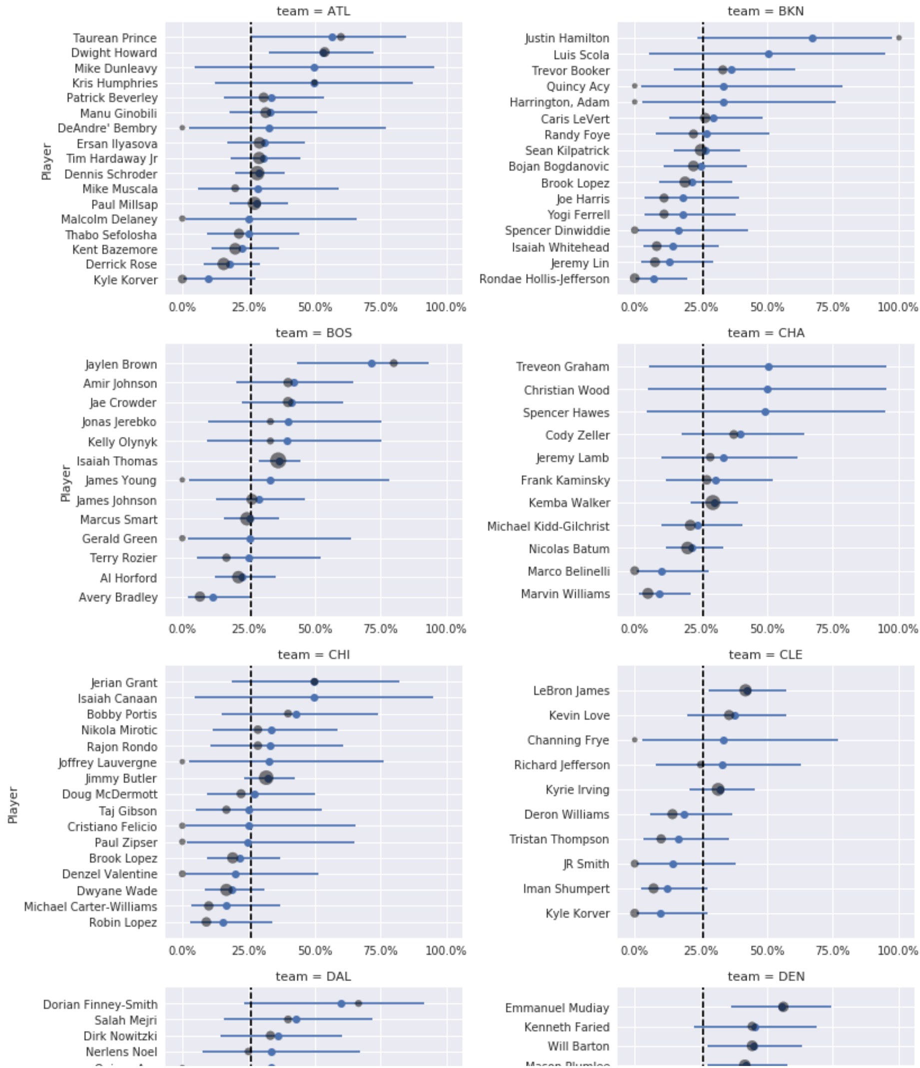
Below we plot the posterior mean and 90% credible interval for p for each player in the data set (grouped by team, for legibility), along with the player's observed foul called percentage when disadvantaged. The area of the point for observed foul called percentage is proportional to the number of plays in which the player was disadvantaged.

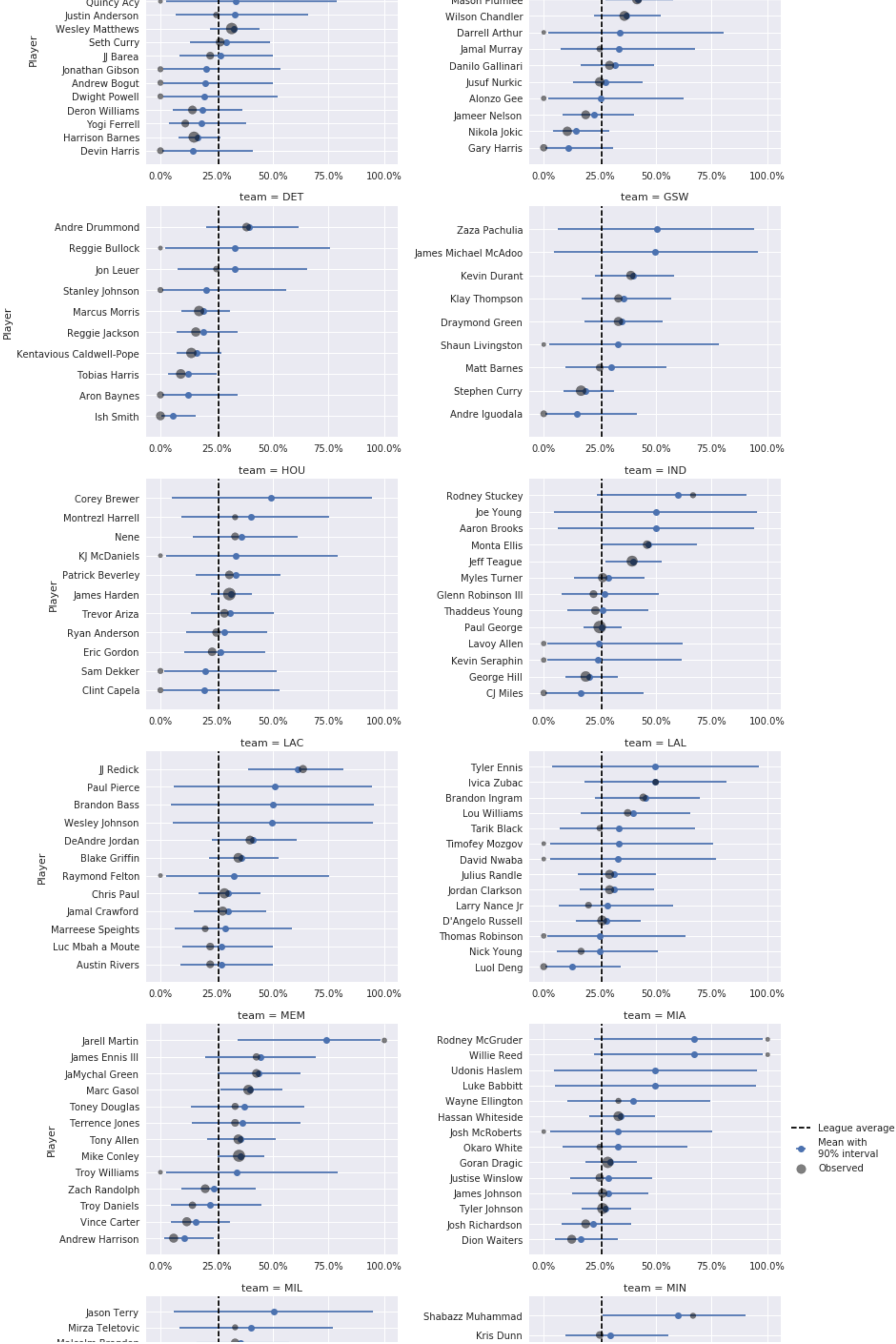
```
def to_param_df(player_df, trace, varnames):
    df = player_df

    for name in varnames:
        mean = trace[name].mean(axis=0)
        low, high = np.percentile(trace[name], [5, 95], axis=0)

        df = df.assign(**{
            '{}_mean'.format(name): mean[df.player_id],
            '{}_low'.format(name): low[df.player_id],
            '{}_high'.format(name): high[df.player_id]
        })

    return df
```

```
bb_df = to_param_df(team_player_map, bb_trace, ['p'])
```

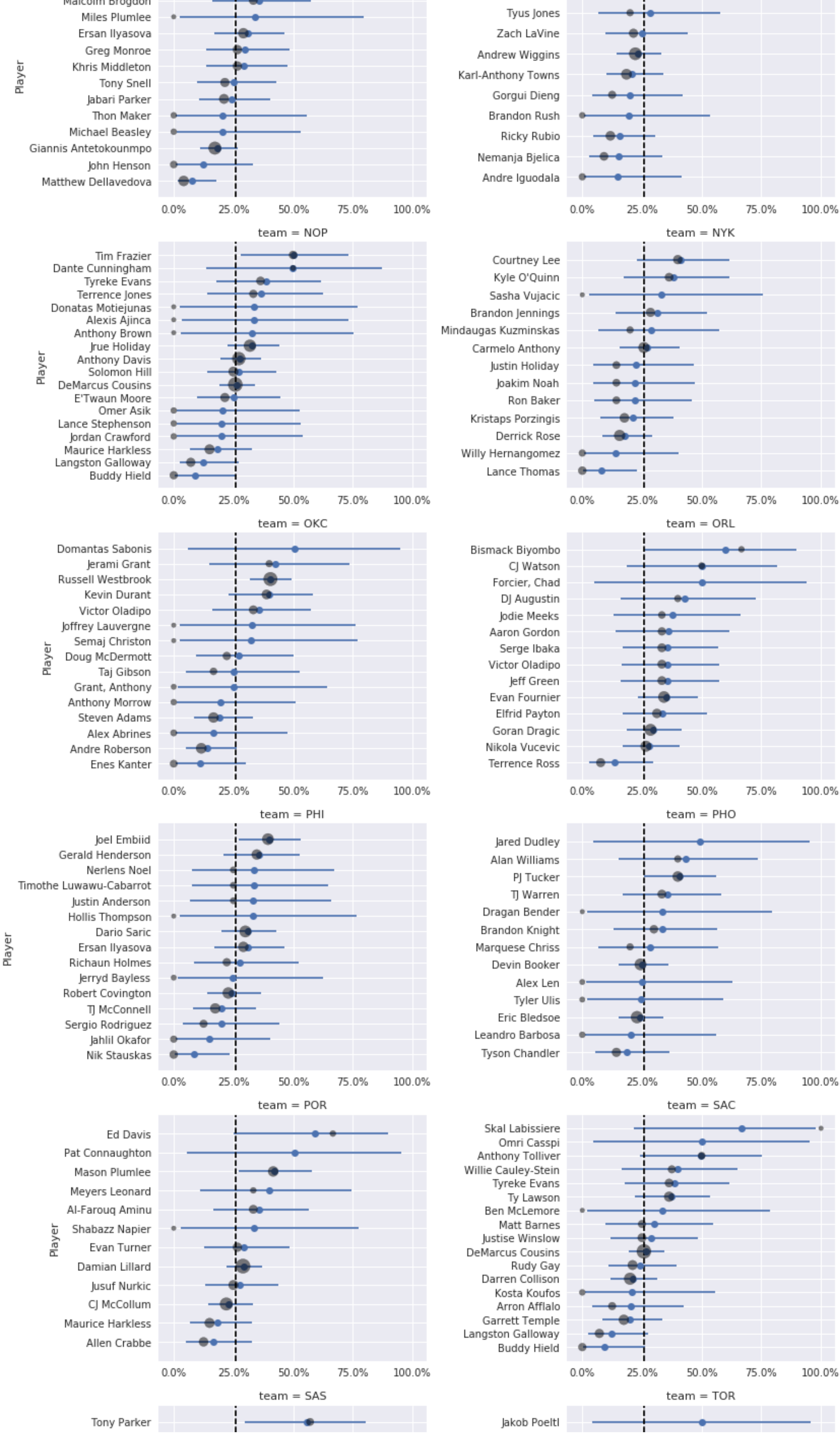
```python
def plot_params(mean, interval, names, ax=None, **kwargs):
    if ax is None:
        fig, ax = plt.subplots(figsize=(8, 6))

    n_players = names.size

    ax.errorbar(mean, np.arange(n_players),
                xerr=np.abs(mean - interval),
                fmt='o',
                label="Mean with\n90% interval")


    ax.set_ylim(-1, n_players)
    ax.set_yticks(np.arange(n_players))
    ax.set_yticklabels(names)

    return ax

def plot_p_params(rate, n_plays, league_mean, ax=None, **kwargs):
    if ax is None:
        ax = plt.gca()

    n_players = rate.size

    ax.scatter(rate, np.arange(n_players),
               c='k', s=20 * np.sqrt(n_plays),
               alpha=0.5, zorder=5,
               label="Observed")

    ax.vlines(league_mean, -1, n_players,
              'k', '--',
              label="League average")

def plot_p_helper(mean, low, high, rate, n_plays, names, league_mean=None, ax=None, **kwargs):
    if ax is None:
        ax = plt.gca()

    mean = mean.values
    rate = rate.values
    n_plays = n_plays.values
    names = names.values

    argsorted_ix = mean.argsort()
    interval = np.row_stack([low, high])

    plot_params(mean[argsorted_ix], interval[:, argsorted_ix], names[argsorted_ix],
                ax=ax, **kwargs)
    plot_p_params(rate[argsorted_ix], n_plays[argsorted_ix], league_mean,
                  ax=ax, **kwargs)
```

```
grid = sns.FacetGrid(bb_df, col='team', col_wrap=2,
                     sharey=False,
                     size=4, aspect=1.5)
grid.map(plot_p_helper,
         'p_mean', 'p_low', 'p_high',
         'disadvantaged_rate', 'disadvantaged_plays', 'name',
         league_mean=obs_rate);

grid.set_axis_labels(r"$p$", "Player");

for ax in grid.axes:
    ax.set_xticks(np.linspace(0, 1, 5));
    ax.set_xticklabels(ax.get_xticklabels(), visible=True)
    ax.xaxis.set_major_formatter(pct_formatter);

grid.fig.tight_layout();
grid.add_legend();
```

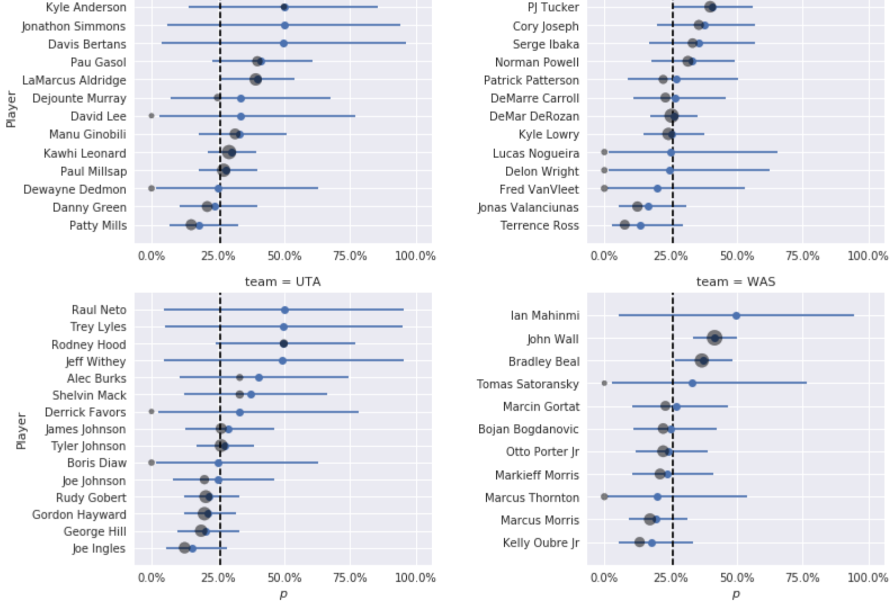




Quincy Acy
Justin Anderson
Wesley Matthews
Seth Curry
JJ Barea
Jonathan Gibson
Andrew Bogut
Dwight Powell
Deron Williams
Yogi Ferrell
Harrison Barnes
Devin Harris
Mason Plumlee
Wilson Chandler
Darrell Arthur
Jamal Murray
Danilo Gallinari
Jusuf Nurkic
Alonzo Gee
Jameer Nelson
Nikola Jokic
Gary Harris
team = DET
Andre Drummond
Reggie Bullock
Jon Leuer
Stanley Johnson
Marcus Morris
Reggie Jackson
Kentavious Caldwell-Pope
Tobias Harris
Aron Baynes
Ish Smith
team = GSW
Zaza Pachulia
James Michael McAdoo
Kevin Durant
Klay Thompson
Draymond Green
Shaun Livingston
Matt Barnes
Stephen Curry
Andre Iguodala
team = HOU
Corey Brewer
Montrezl Harrell
Nene
KJ McDaniels
Patrick Beverley
James Harden
Trevor Ariza
Ryan Anderson
Eric Gordon
Sam Dekker
Clint Capela
team = IND
Rodney Stuckey
Joe Young
Aaron Brooks
Monta Ellis
Jeff Teague
Myles Turner
Glenn Robinson III
Thaddeus Young
Paul George
Lavoy Allen
Kevin Seraphin
George Hill
CJ Miles
team = LAC
JJ Redick
Paul Pierce
Brandon Bass
Wesley Johnson
DeAndre Jordan
Blake Griffin
Raymond Felton
Chris Paul
Jamal Crawford
Marreese Speights
Luc Mbah a Moute
Austin Rivers
team = LAL
Tyler Ennis
Ivica Zubac
Brandon Ingram
Lou Williams
Tarik Black
Timofey Mozgov
David Nwaba
Julius Randle
Jordan Clarkson
Larry Nance Jr
D'Angelo Russell
Thomas Robinson
Nick Young
Luol Deng
team = MEM
Jarell Martin
James Ennis III
JaMychal Green
Marc Gasol
Toney Douglas
Terrence Jones
Tony Allen
Mike Conley
Troy Williams
Zach Randolph
Troy Daniels
Vince Carter
Andrew Harrison
team = MIA
Rodney McGruder
Willie Reed
Udonis Haslem
Luke Babbitt
Wayne Ellington
Hassan Whiteside
Josh McRoberts
Okaro White
Goran Dragic
Justise Winslow
James Johnson
Tyler Johnson
Josh Richardson
Dion Waiters
team = MIL
Jason Terry
Mirza Teletovic
Malcolm Brogdon
team = MIN
Shabazz Muhammad
Kris Dunn
Player
0.0%
25.0%
50.0%
75.0%
100.0%
League average
Mean with 90% interval
Observed



Malcolm Brogdon
Miles Plumlee
Ersan Ilyasova
Greg Monroe
Khris Middleton
Tony Snell
Jabari Parker
Thon Maker
Michael Beasley
Giannis Antetokounmpo
John Henson
Matthew Dellavedova
Player
Tyus Jones
Zach LaVine
Andrew Wiggins
Karl-Anthony Towns
Gorgui Dieng
Brandon Rush
Ricky Rubio
Nemanja Bjelica
Andre Iguodala
0.0%
25.0%
50.0%
75.0%
100.0%
team = NOP
Tim Frazier
Dante Cunningham
Tyreke Evans
Terrence Jones
Donatas Motiejunas
Alexis Ajinca
Anthony Brown
Jrue Holiday
Anthony Davis
Solomon Hill
DeMarcus Cousins
E'Twaun Moore
Omer Asik
Lance Stephenson
Jordan Crawford
Maurice Harkless
Langston Galloway
Buddy Hield
team = NYK
Courtney Lee
Kyle O'Quinn
Sasha Vujacic
Brandon Jennings
Mindaugas Kuzminskas
Carmelo Anthony
Justin Holiday
Joakim Noah
Ron Baker
Kristaps Porzingis
Derrick Rose
Willy Hernangomez
Lance Thomas
team = OKC
Domantas Sabonis
Jerami Grant
Russell Westbrook
Kevin Durant
Victor Oladipo
Joffrey Lauvergne
Semaj Christon
Doug McDermott
Taj Gibson
Grant, Anthony
Anthony Morrow
Steven Adams
Alex Abrines
Andre Roberson
Enes Kanter
team = ORL
Bismack Biyombo
CJ Watson
Forcier, Chad
DJ Augustin
Jodie Meeks
Aaron Gordon
Serge Ibaka
Victor Oladipo
Jeff Green
Evan Fournier
Elfrid Payton
Goran Dragic
Nikola Vucevic
Terrence Ross
team = PHI
Joel Embiid
Gerald Henderson
Nerlens Noel
Timothe Luwawu-Cabarrot
Justin Anderson
Hollis Thompson
Dario Saric
Ersan Ilyasova
Richaun Holmes
Jerryd Bayless
Robert Covington
TJ McConnell
Sergio Rodriguez
Jahlil Okafor
Nik Stauskas
team = PHO
Jared Dudley
Alan Williams
PJ Tucker
TJ Warren
Dragan Bender
Brandon Knight
Marquese Chriss
Devin Booker
Alex Len
Tyler Ulis
Eric Bledsoe
Leandro Barbosa
Tyson Chandler
team = POR
Ed Davis
Pat Connaughton
Mason Plumlee
Meyers Leonard
Al-Farouq Aminu
Shabazz Napier
Evan Turner
Damian Lillard
Jusuf Nurkic
CJ McCollum
Maurice Harkless
Allen Crabbe
team = SAC
Skal Labissiere
Omri Casspi
Anthony Tolliver
Willie Cauley-Stein
Tyreke Evans
Ty Lawson
Ben McLemore
Matt Barnes
Justise Winslow
DeMarcus Cousins
Rudy Gay
Darren Collison
Kosta Koufos
Arron Afflalo
Garrett Temple
Langston Galloway
Buddy Hield
team = SAS
Tony Parker
team = TOR
Jakob Poeltl



These plots reveal an undesirable property of this model and its inferences. Since the prior distribution on p_i is uniform on the interval $[0, 1]$, all posterior estimates of p_i are pulled towards the prior expected value of 50%. This phenomenon is known as shrinkage. In extreme cases of players that were never disadvantaged, the posterior estimate of p_i is quite close to 50%. For these players, the league average foul call rate would seem to be a much more reasonable estimate of p_i than 50%. The league average foul call rate is shown as a dotted black line on the charts above.

There are several possible modifications of the beta-Bernoulli model that can cause shrinkage toward the league average. Perhaps the most straightforward is the empirical Bayesian method (https://en.wikipedia.org/wiki/Empirical_Bayes_method) that sets the parameters of the prior distribution on p_i using the observed data. In this framework, there are many methods of choosing prior hyperparameters that make the prior expected value equal to the league average, therefore causing shrinkage toward the league average. We do not use empirical Bayesian methods in this post as they make it cumbersome to build the more complex models we want to use to understand the relationship between salary and foul calls. Empirical Bayesian methods are, however, an approximation to the fully hierachical models we begin building in the next section.

Hierarchical logistic-normal model

A hierarchical logistic-normal (https://en.wikipedia.org/wiki/Logit-normal_distribution) model addresses some of the shortcomings of the beta-Bernoulli model. For simplicity, this model focuses exclusively on the disadvantaged player and assumes that the log-odds (https://en.wikipedia.org/wiki/Logit) of a foul call for a given disadvantaged player are normally distributed. That is,

$$\log\left(\frac{p_i}{1 - p_i}\right) \sim N(\mu, \sigma^2)$$

$$\eta_k = \log\left(\frac{p_{i(k)}}{1 - p_{i(k)}}\right),$$

which is equivalent to

$$p_{i(k)} = \frac{1}{1 + \exp(-\eta_k)}.$$

We address the beta-Bernoulli model's shrinkage problem by placing a normal hyperprior distribution on μ, $\mu \sim N(0, 100)$. This shared hyperprior makes this model hierarchical. To complete the specification of this model, we place a half-Cauchy (https://en.wikipedia.org/wiki/Cauchy_distribution) prior on σ, $\sigma \sim \text{HalfCauchy}(2.5)$.

```
with pm.Model() as ln_model:
    μ = pm.Normal('μ', 0., 10.)
    Δ = pm.Normal('Δ', 0., 1., shape=n_players)
    σ = pm.HalfCauchy('σ', 2.5)

    p_player = pm.Deterministic('p_player', pm.math.sigmoid(μ + Δ * σ))

    η = μ + Δ[disadvantaged_id] * σ
    p = pm.Deterministic('p', pm.math.sigmoid(η))

    y = pm.Bernoulli('y_obs', p, observed=foul_called)
```

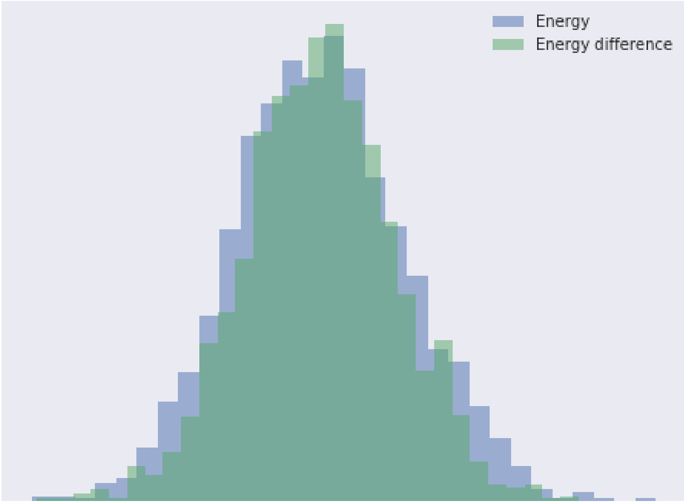
Throughout this post we use an offset parameterization (http://twiecki.github.io/blog/2017/02/08/bayesian-hierchical-non-centered/) for hierarchical models that significantly improves sampling efficiency. We now sample from this model.

```
ln_trace = sample(ln_model, N_TUNE, N_SAMPLES, SEED)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using advi...
  9%|█          | 17001/200000 [00:12<02:13, 1372.87it/s]Median ELBO converged.
Finished [100%]: Average ELBO = -3,520.3

100%|██████████| 4000/4000 [02:27<00:00, 65.27it/s]
```

The energy plot for this model gives no cause for concern.

```
energy_plot(ln_trace)
```

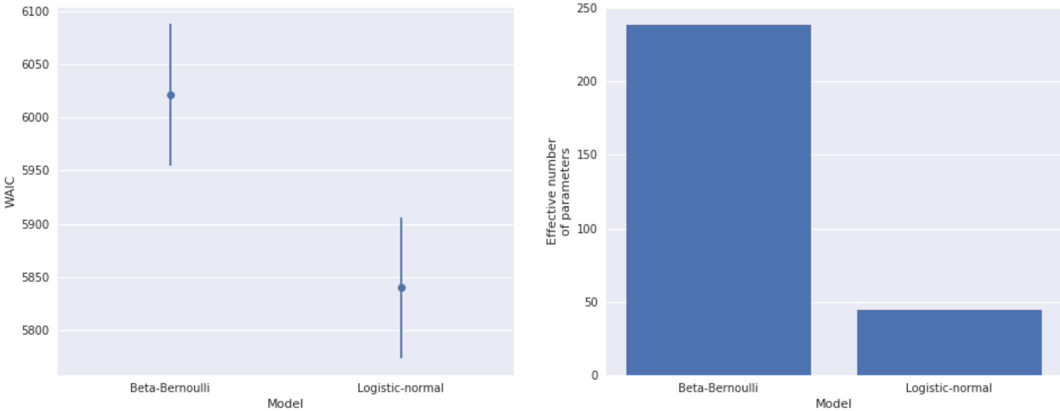


We now calculate the WAIC of the logistic-normal model, and compare it to that of the beta-Bernoulli model.

```
waic_df = waic_df.append(get_waic_df(ln_model, ln_trace, "Logistic-normal"))
```

```
def waic_plot(waic_df):
    fig, (waic_ax, p_ax) = plt.subplots(ncols=2, sharex=True, figsize=(16, 6))

    waic_x = np.arange(waic_df.shape[0])

    waic_ax.errorbar(waic_x, waic_df.WAIC,
                     yerr=waic_df.WAIC_se,
                     fmt='o')

    waic_ax.set_xticks(waic_x)
    waic_ax.xaxis.grid(False)
    waic_ax.set_xticklabels(waic_df.index)
    waic_ax.set_xlabel("Model")

    waic_ax.set_ylabel("WAIC")

    p_ax.bar(waic_x, waic_df.p_WAIC)

    p_ax.xaxis.grid(False)
    p_ax.set_xlabel("Model")

    p_ax.set_ylabel("Effective number\nof parameters")
```

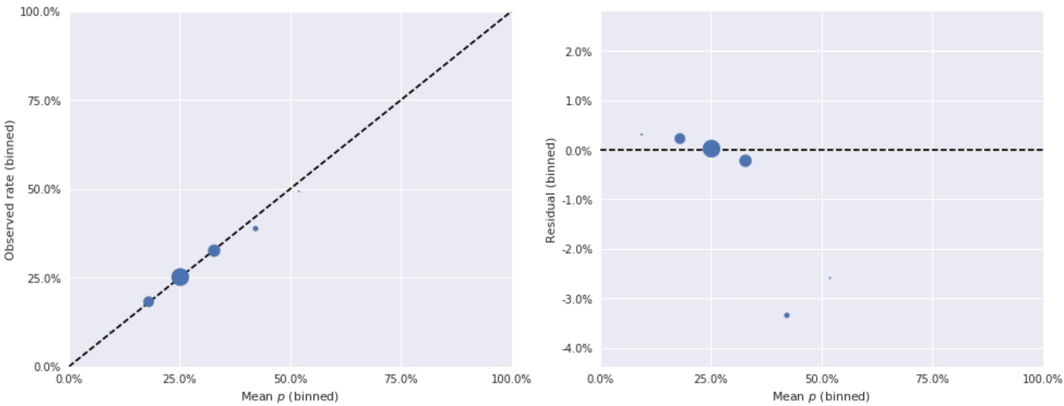
```
waic_plot(waic_df)
```



The left-hand plot above shows that the logistic-normal model is a significant improvement in WAIC over the beta-Bernoulli model, which is unsurprising. The right-hand plot shows that the logistic-normal model has roughly 20% the number of effective parameters as the beta-Bernoulli model. This reduction is due to the partial pooling effect of the hierarchical prior. The hyperprior on μ causes observations for one player to impact the estimate of p_i for all players; this sharing of information across players is responsible for the large decrease in the number of effective parameters.

Finally, we examine the binned residuals for the logistic-normal model.

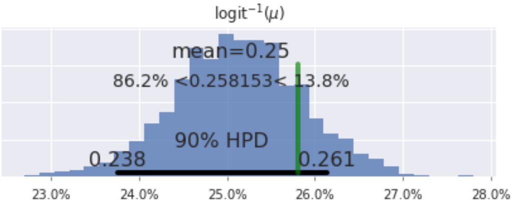
```
bin_obs, bin_p, bin_counts = binned_residuals(foul_called, ln_trace['p'])

binned_residual_plot(bin_obs, bin_p, bin_counts)
```



These binned residuals are much smaller than those of the beta-Bernoulli model, which is further confirmation that the logistic-normal model is preferable.

Below we plot the posterior distribution of $\text{logit}^{-1}(\mu)$, and we see that the observed foul call rate of approximately 25.1% lies within its 90% interval.

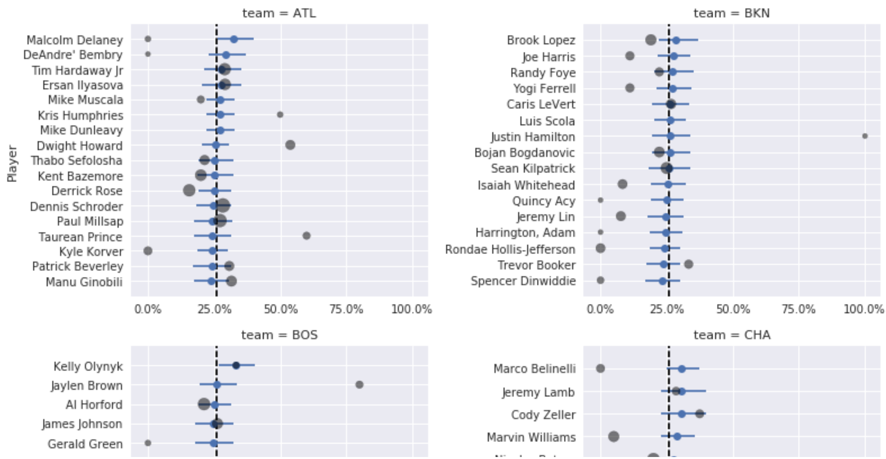
```
ax, = pm.plot_posterior(ln_trace, ['μ'],
                        alpha_level=0.1, transform=expit, ref_val=obs_rate,
                        lw=0., alpha=0.75)

ax.xaxis.set_major_formatter(pct_formatter);

ax.set_title(r"$\operatorname{logit}^{-1}(\mu)$");
```

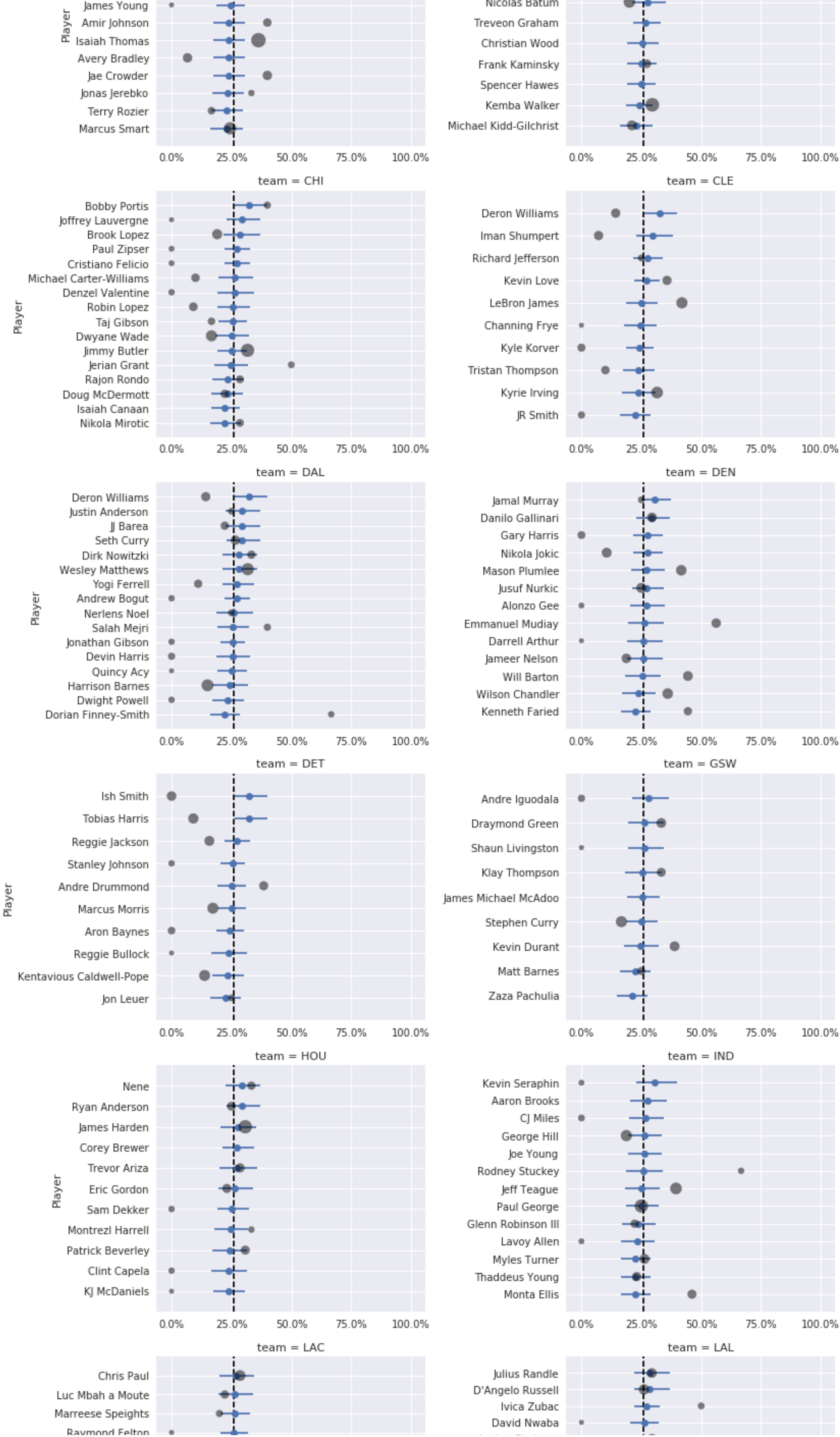


With this posterior for $\operatorname{logit}^{-1}(\mu)$, we see the desired posterior shrinkage of each p_i toward the observed foul call rate.

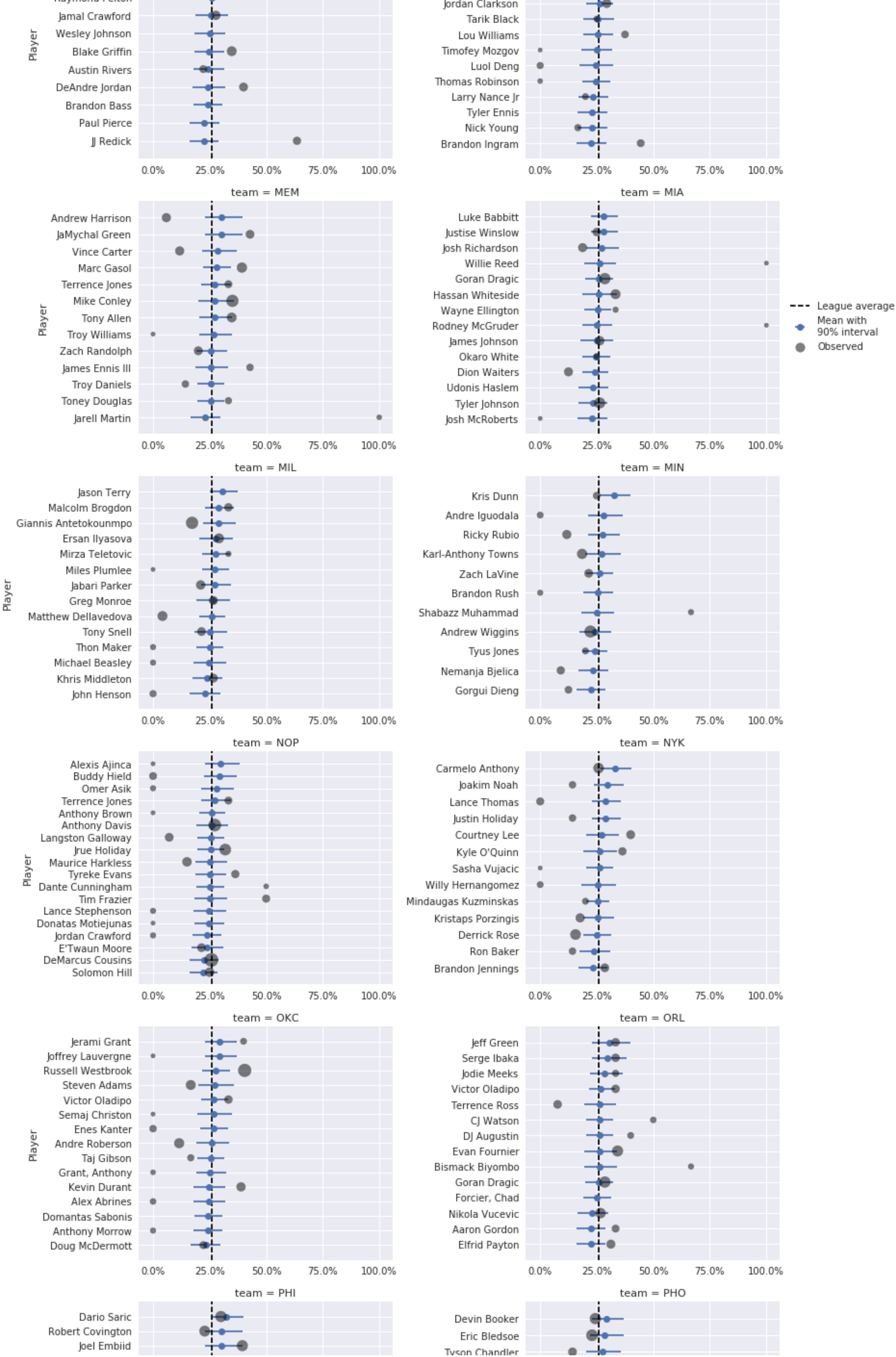
```
ln_df = to_param_df(team_player_map, ln_trace, ['p'])
```

```
grid = sns.FacetGrid(ln_df, col='team', col_wrap=2,
                     sharey=False,
                     size=4, aspect=1.5)
grid.map(plot_p_helper,
         'p_mean', 'p_low', 'p_high',
         'disadvantaged_rate', 'disadvantaged_plays', 'name',
         league_mean=obs_rate);

grid.set_axis_labels(r"$p$", "Player");

for ax in grid.axes:
    ax.set_xticks(np.linspace(0, 1, 5));
    ax.set_xticklabels(ax.get_xticklabels(), visible=True)
    ax.xaxis.set_major_formatter(pct_formatter);

grid.fig.tight_layout();
grid.add_legend();
```

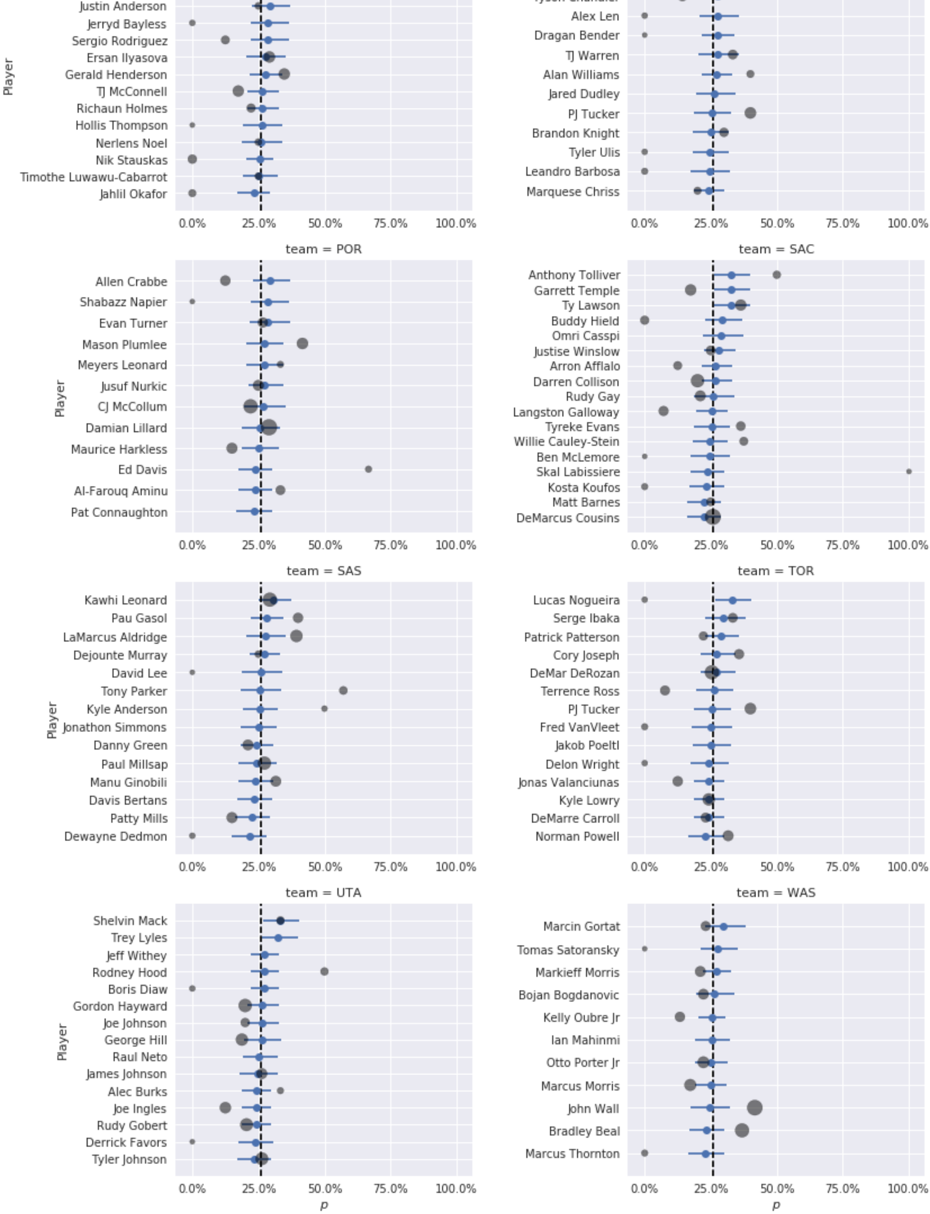




Player
James Young
Amir Johnson
Isaiah Thomas
Avery Bradley
Jae Crowder
Jonas Jerebko
Terry Rozier
Marcus Smart
Nicolas Batum
Treveon Graham
Christian Wood
Frank Kaminsky
Spencer Hawes
Kemba Walker
Michael Kidd-Gilchrist
0.0%
25.0%
50.0%
75.0%
100.0%
team = CHI
Bobby Portis
Joffrey Lauvergne
Brook Lopez
Paul Zipser
Cristiano Felicio
Michael Carter-Williams
Denzel Valentine
Robin Lopez
Taj Gibson
Dwyane Wade
Jimmy Butler
Jerian Grant
Rajon Rondo
Doug McDermott
Isaiah Canaan
Nikola Mirotic
team = CLE
Deron Williams
Iman Shumpert
Richard Jefferson
Kevin Love
LeBron James
Channing Frye
Kyle Korver
Tristan Thompson
Kyrie Irving
JR Smith
team = DAL
Deron Williams
Justin Anderson
JJ Barea
Seth Curry
Dirk Nowitzki
Wesley Matthews
Yogi Ferrell
Andrew Bogut
Nerlens Noel
Salah Mejri
Jonathan Gibson
Devin Harris
Quincy Acy
Harrison Barnes
Dwight Powell
Dorian Finney-Smith
team = DEN
Jamal Murray
Danilo Gallinari
Gary Harris
Nikola Jokic
Mason Plumlee
Jusuf Nurkic
Alonzo Gee
Emmanuel Mudiay
Darrell Arthur
Jameer Nelson
Will Barton
Wilson Chandler
Kenneth Faried
team = DET
Ish Smith
Tobias Harris
Reggie Jackson
Stanley Johnson
Andre Drummond
Marcus Morris
Aron Baynes
Reggie Bullock
Kentavious Caldwell-Pope
Jon Leuer
team = GSW
Andre Iguodala
Draymond Green
Shaun Livingston
Klay Thompson
James Michael McAdoo
Stephen Curry
Kevin Durant
Matt Barnes
Zaza Pachulia
team = HOU
Nene
Ryan Anderson
James Harden
Corey Brewer
Trevor Ariza
Eric Gordon
Sam Dekker
Montrezl Harrell
Patrick Beverley
Clint Capela
KJ McDaniels
team = IND
Kevin Seraphin
Aaron Brooks
CJ Miles
George Hill
Joe Young
Rodney Stuckey
Jeff Teague
Paul George
Glenn Robinson III
Lavoy Allen
Myles Turner
Thaddeus Young
Monta Ellis
team = LAC
Chris Paul
Luc Mbah a Moute
Marreese Speights
Raymond Felton
team = LAL
Julius Randle
D'Angelo Russell
Ivica Zubac
David Nwaba

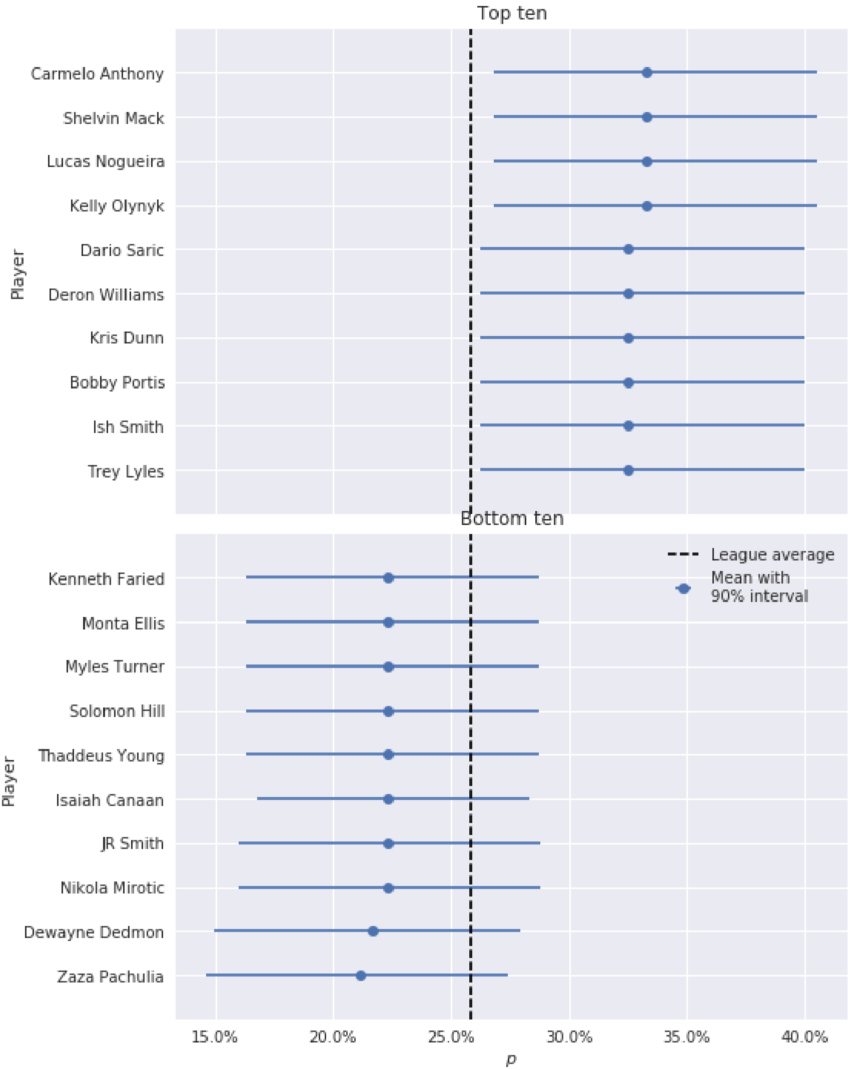


Jamal Crawford
Wesley Johnson
Blake Griffin
Austin Rivers
DeAndre Jordan
Brandon Bass
Paul Pierce
JJ Redick
Player
0.0%
25.0%
50.0%
75.0%
100.0%
Jordan Clarkson
Tarik Black
Lou Williams
Timofey Mozgov
Luol Deng
Thomas Robinson
Larry Nance Jr
Tyler Ennis
Nick Young
Brandon Ingram
team = MEM
Andrew Harrison
JaMychal Green
Vince Carter
Marc Gasol
Terrence Jones
Mike Conley
Tony Allen
Troy Williams
Zach Randolph
James Ennis III
Troy Daniels
Toney Douglas
Jarell Martin
team = MIA
Luke Babbitt
Justise Winslow
Josh Richardson
Willie Reed
Goran Dragic
Hassan Whiteside
Wayne Ellington
Rodney McGruder
James Johnson
Okaro White
Dion Waiters
Udonis Haslem
Tyler Johnson
Josh McRoberts
League average
Mean with 90% interval
Observed
team = MIL
Jason Terry
Malcolm Brogdon
Giannis Antetokounmpo
Ersan Ilyasova
Mirza Teletovic
Miles Plumlee
Jabari Parker
Greg Monroe
Matthew Dellavedova
Tony Snell
Thon Maker
Michael Beasley
Khris Middleton
John Henson
team = MIN
Kris Dunn
Andre Iguodala
Ricky Rubio
Karl-Anthony Towns
Zach LaVine
Brandon Rush
Shabazz Muhammad
Andrew Wiggins
Tyus Jones
Nemanja Bjelica
Gorgui Dieng
team = NOP
Alexis Ajinca
Buddy Hield
Omer Asik
Terrence Jones
Anthony Brown
Anthony Davis
Langston Galloway
Jrue Holiday
Maurice Harkless
Tyreke Evans
Dante Cunningham
Tim Frazier
Lance Stephenson
Donatas Motiejunas
Jordan Crawford
E'Twaun Moore
DeMarcus Cousins
Solomon Hill
team = NYK
Carmelo Anthony
Joakim Noah
Lance Thomas
Justin Holiday
Courtney Lee
Kyle O'Quinn
Sasha Vujacic
Willy Hernangomez
Mindaugas Kuzminskas
Kristaps Porzingis
Derrick Rose
Ron Baker
Brandon Jennings
team = OKC
Jerami Grant
Joffrey Lauvergne
Russell Westbrook
Steven Adams
Victor Oladipo
Semaj Christon
Enes Kanter
Andre Roberson
Taj Gibson
Grant, Anthony
Kevin Durant
Alex Abrines
Domantas Sabonis
Anthony Morrow
Doug McDermott
team = ORL
Jeff Green
Serge Ibaka
Jodie Meeks
Victor Oladipo
Terrence Ross
CJ Watson
DJ Augustin
Evan Fournier
Bismack Biyombo
Goran Dragic
Forcier, Chad
Nikola Vucevic
Aaron Gordon
Elfrid Payton
team = PHI
Dario Saric
Robert Covington
Joel Embiid
team = PHO
Devin Booker
Eric Bledsoe
Tyson Chandler



The inferences in these plots are markedly different from those of the beta-Bernoulli model. Most strikingly, we see that estimates have been shrunk towards the league average foul call rate, and that players that were never disadvantaged have posterior foul call probabilities quite close to that rate. As a consequence of this more reasonable shrinkage, the range of values taken by the posterior p_i estimates is much smaller for the logistic-normal model than for the beta-Bernoulli model. Below we plot the top- and bottom-ten players by p_i.

```
fig, (top_ax, bottom_ax) = plt.subplots(nrows=2, sharex=True, figsize=(8, 10))

by_p = (ln_df.drop_duplicates(['player_id'])
             .sort_values('p_mean'))
p_top = by_p.iloc[-10:]

plot_params(p_top.p_mean.values, p_top[['p_low', 'p_high']].values.T,
            p_top.name.values,
            ax=top_ax);
top_ax.vlines(obs_rate, -1, 10,
              'k', '--',
              label=r"League average");

top_ax.xaxis.set_major_formatter(pct_formatter);

top_ax.set_ylabel("Player");

top_ax.set_title("Top ten");

p_bottom = by_p.iloc[:10]

plot_params(p_bottom.p_mean.values, p_bottom[['p_low', 'p_high']].values.T,
            p_bottom.name.values,
            ax=bottom_ax);
bottom_ax.vlines(obs_rate, -1, 10,
                 'k', '--',
                 label=r"League average");

bottom_ax.xaxis.set_major_formatter(pct_formatter);
bottom_ax.set_xlabel(r"$p$");

bottom_ax.set_ylabel("Player");

fig.tight_layout();
bottom_ax.legend(loc=1);
bottom_ax.set_title("Bottom ten");
```



Item-response (Rasch) model

The hierarchical logistic-normal model is certainly an improvement over the beta-Bernoulli model, but both of these models have focused solely on the disadvantaged player. It seems quite important to understand the contribution of not just the disadvantaged player, but also of the committing player in each play to the probability of a foul call. Item-response theory (https://en.wikipedia.org/wiki/Item_response_theory) (IRT) provides generalizations the logistic-normal model that can account for the influence of both players involved in a play. IRT originated in psychometrics as a way to simultaneously measure individual aptitude and question difficulty based on test-response data, and has subsequently found many other applications. We use IRT to model foul calls by considering disadvantaged players as analagous to test takers and committing players as analagous to questions. Specifically, we will use the Rasch model (https://en.wikipedia.org/wiki/Rasch_model) for the probability $p_{i,j}$, that a foul is called on a play where player i is disadvantaged by committing player j. This model posits that each player has a latent ability, θ_i, that governs how often fouls are called when they are disadvantaged and a latent difficulty b_j that governs how often fouls are not called when they are committing. The probability that a foul is called on a play where player i is disadvantaged and player j is committing is then a function of the difference between the corresponding latent ability and difficulty parameters,

$$\begin{aligned}\eta_k &= \theta_{i(k)} - b_{j(k)} \\ p_k &= \frac{1}{1 + \exp(-\eta_k)}.\end{aligned}$$

In this model, a player with a large value of θ_i is more likely to get a foul called when they are disadvantaged, and a player with a large value of b_j is less likely to have a foul called when they are committing. If $\theta_{i(k)} = b_{j(k)}$, there is a 50% chance a foul is called on that play.

To complete the specification of this model, we place priors on θ_i and b_j. Similarly to η in the logistic-normal model, we place a hierarchical normal prior on θ_i,

$$\begin{aligned}\mu_\theta &\sim N(0, 100) \\ \sigma_\theta &\sim \text{HalfCauchy}(2.5) \\ \theta_i &\sim N(\mu_\theta, \sigma_\theta^2).\end{aligned}$$

```
with pm.Model() as rasch_model:
    μ_θ = pm.Normal('μ_θ', 0., 10.)
    Δ_θ = pm.Normal('Δ_θ', 0., 1., shape=n_players)
    σ_θ = pm.HalfCauchy('σ_θ', 2.5)
    θ = pm.Deterministic('θ', μ_θ + Δ_θ * σ_θ)
```

We also place a hierarchical normal prior on b_j, though this prior must be subtley different from that on θ_i. Since θ_i and b_j are latent variables, there is no natural scale on which they should be measured. If each θ_i and b_j are shifted by the same amount, say δ, the likelihood does not change. That is, if $\tilde{\theta}_i = \theta_i + \delta$ and $\tilde{b}_j = b_j + \delta$, then

$$\tilde{\eta}_{i,j} = \tilde{\theta}_i - \tilde{b}_j = \theta_i + \delta - (b_j + \delta) = \theta_i - b_j = \eta_{i,j}.$$

Therefore, if we allow θ_i and β_j to be shifted by arbitrary amounts, the Rasch model is not identified (https://en.wikipedia.org/wiki/Identifiability). We identify the Rasch model by constraining the mean of the hyperprior on b_j to be zero,

$$\begin{aligned}\sigma_b &\sim \text{HalfCauchy}(2.5) \\ b_j &\sim N(0, \sigma_b^2).\end{aligned}$$

```
with rasch_model:
    Δ_b = pm.Normal('Δ_b', 0., 1., shape=n_players)
    σ_b = pm.HalfCauchy('σ_b', 2.5)
    b = pm.Deterministic('b', Δ_b * σ_b)
```

We now specify the Rasch model's likelihood and sample from it.

```
committing_id = df.committing_id.values
```

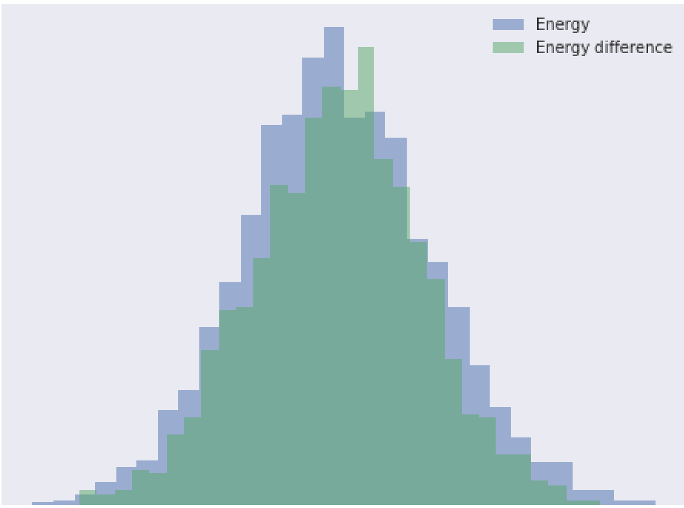
```
with rasch_model:
    η = θ[disadvantaged_id] - b[committing_id]
    p = pm.Deterministic('p', pm.math.sigmoid(η))

    y = pm.Bernoulli('y_obs', p, observed=foul_called)
```

```
rasch_trace = sample(rasch_model, N_TUNE, N_SAMPLES, SEED)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using advi...
Average ELBO = -3,729.5:  11%|█          | 22583/200000 [00:19<02:33, 1156.17it/s]Median ELBO c
onverged.
Finished [100%]: Average ELBO = -3,037.1

100%|██████████| 4000/4000 [02:01<00:00, 32.81it/s]
```

Again, the energy plot for this model gives no cause for concern.

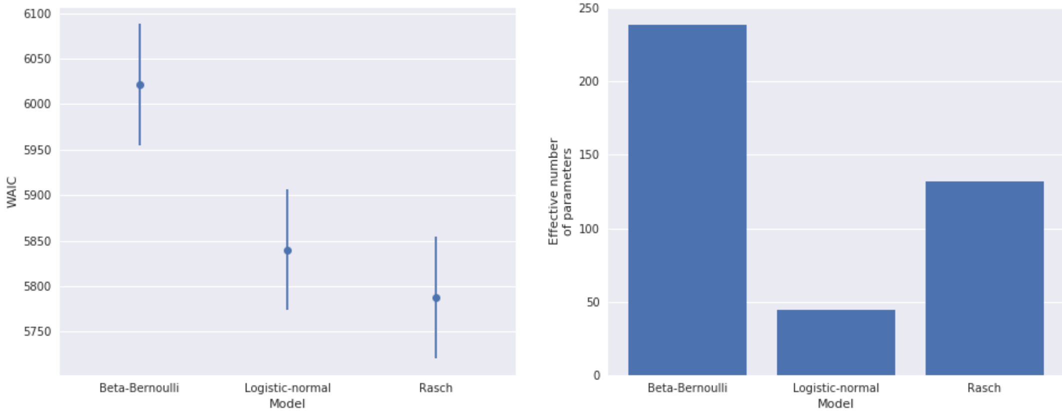
```
energy_plot(rasch_trace)
```



Below we show the WAIC of our three models.

```
waic_df = waic_df.append(get_waic_df(rasch_model, rasch_trace, "Rasch"))
```

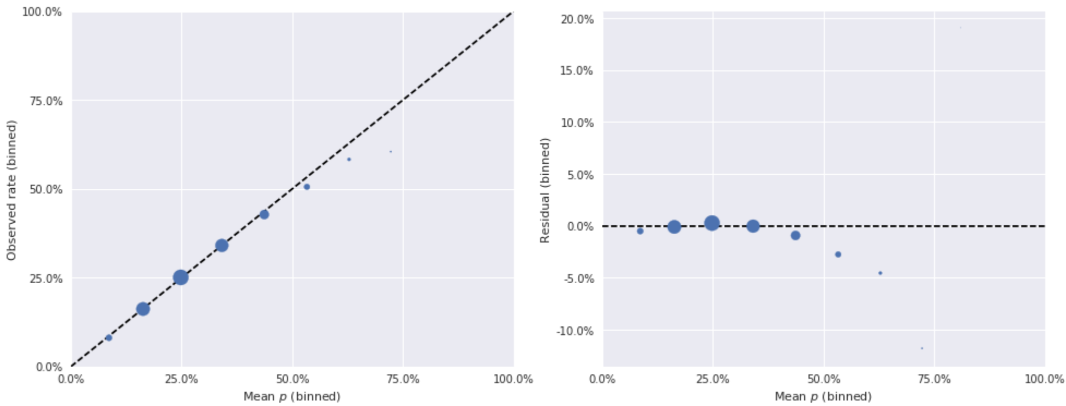
```
waic_plot(waic_df)
```



The Rasch model represents a moderate WAIC improvement over the logistic-normal model, and unsurprisingly has many more effective parameters (since it added a nominal parameter, b_j, per player).

The Rasch model also has reasonable binned residuals, with very few events having residuals above 5%.

```
bin_obs, bin_p, bin_counts = binned_residuals(foul_called, rasch_trace['p'])

binned_residual_plot(bin_obs, bin_p, bin_counts)
```

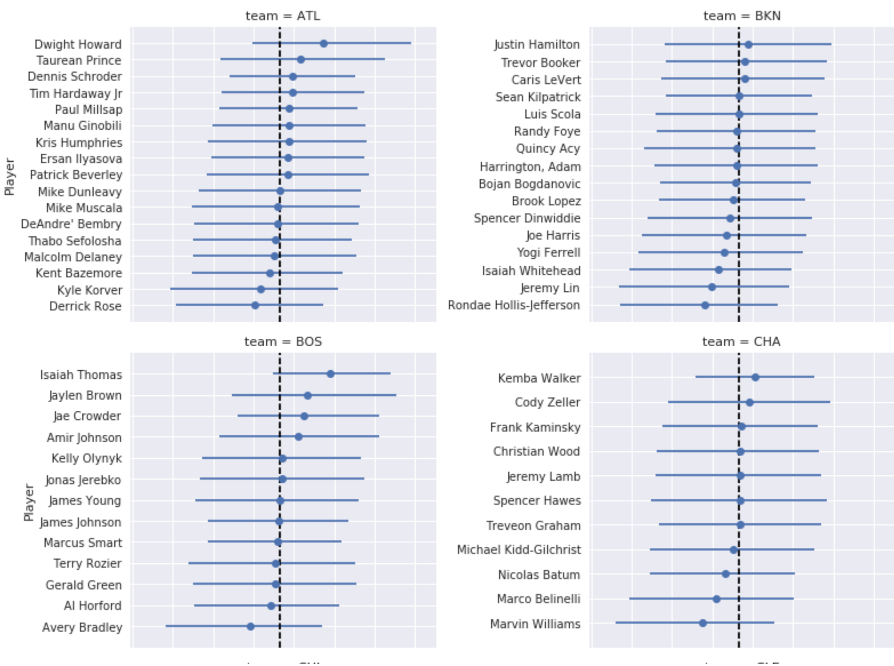


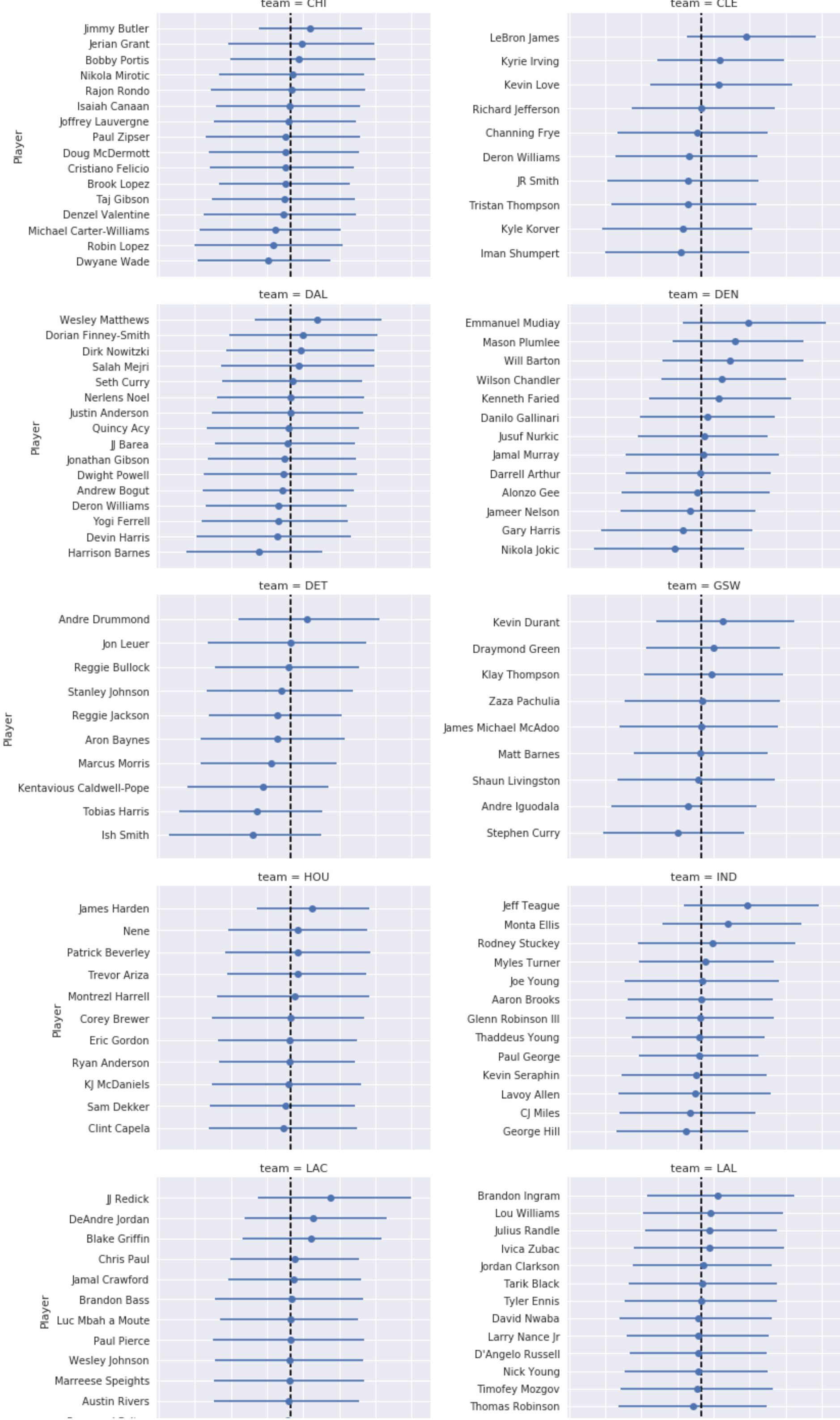
For the Rasch model (and subsequent models), we switch from visualizing the per-player call probabilities to the latent parameters θ_i and b_j.

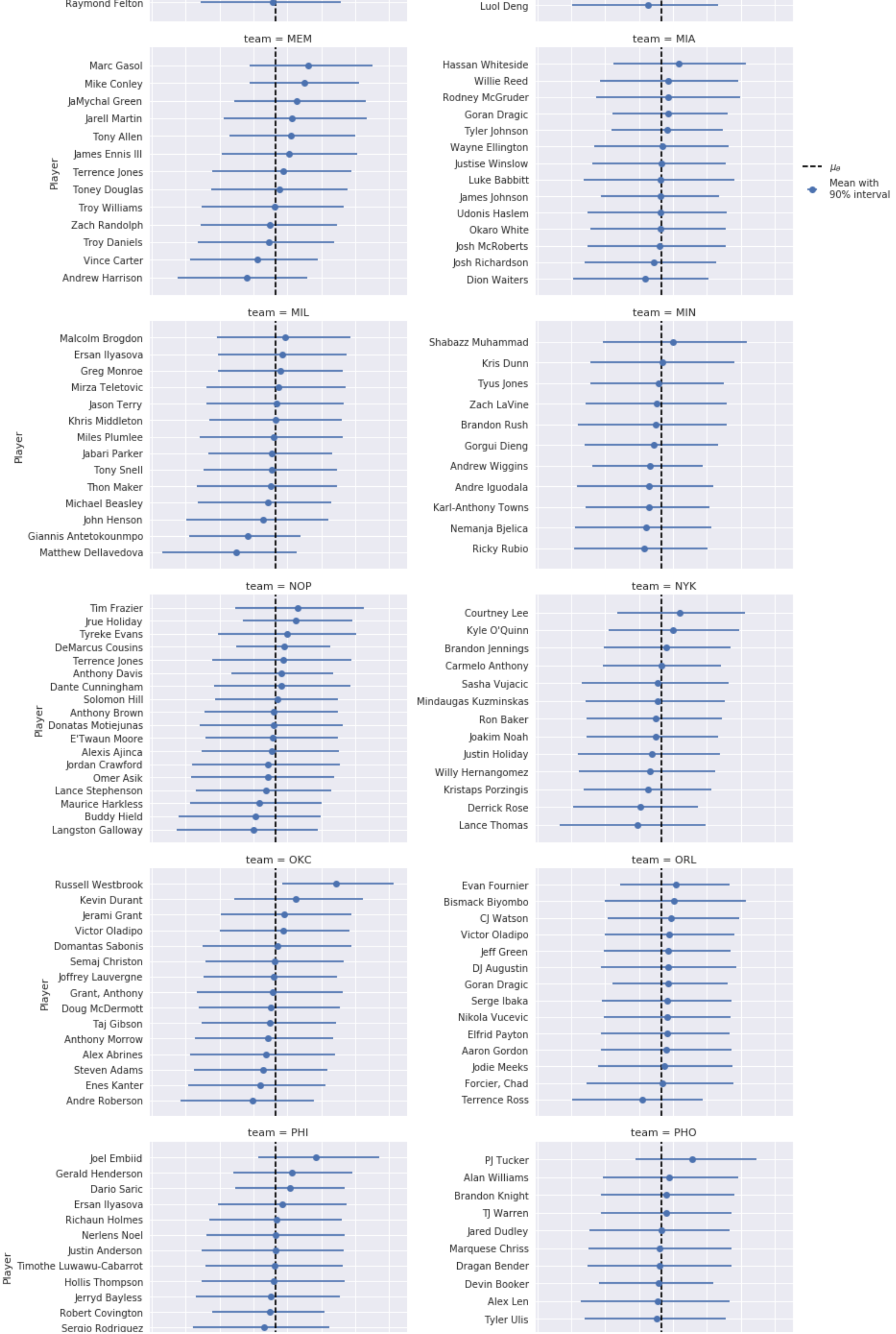
```
μ_θ_mean = rasch_trace['μ_θ'].mean()

rasch_df = to_param_df(team_player_map, rasch_trace, ['θ', 'b'])
```

```
def plot_params_helper(mean, low, high, names, league_mean=None, league_mean_name=None, ax=Non
e, **kwargs):
    if ax is None:
        ax = plt.gca()

    mean = mean.values
    names = names.values

    argsorted_ix = mean.argsort()
    interval = np.row_stack([low, high])

    plot_params(mean[argsorted_ix], interval[:, argsorted_ix], names[argsorted_ix],
                ax=ax, **kwargs)

    if league_mean is not None:
        ax.vlines(league_mean, -1, names.size,
                  'k', '--',
                  label=league_mean_name)
```

```
grid = sns.FacetGrid(rasch_df, col='team', col_wrap=2,
                     sharey=False,
                     size=4, aspect=1.5)
grid.map(plot_params_helper,
         'θ_mean', 'θ_low', 'θ_high', 'name',
         league_mean=μ_θ_mean,
         league_mean_name=r"$\mu_{\theta}$");

grid.set_axis_labels(r"$\theta$", "Player");

grid.fig.tight_layout();
grid.add_legend();
```

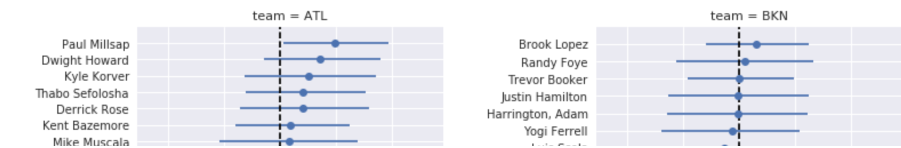


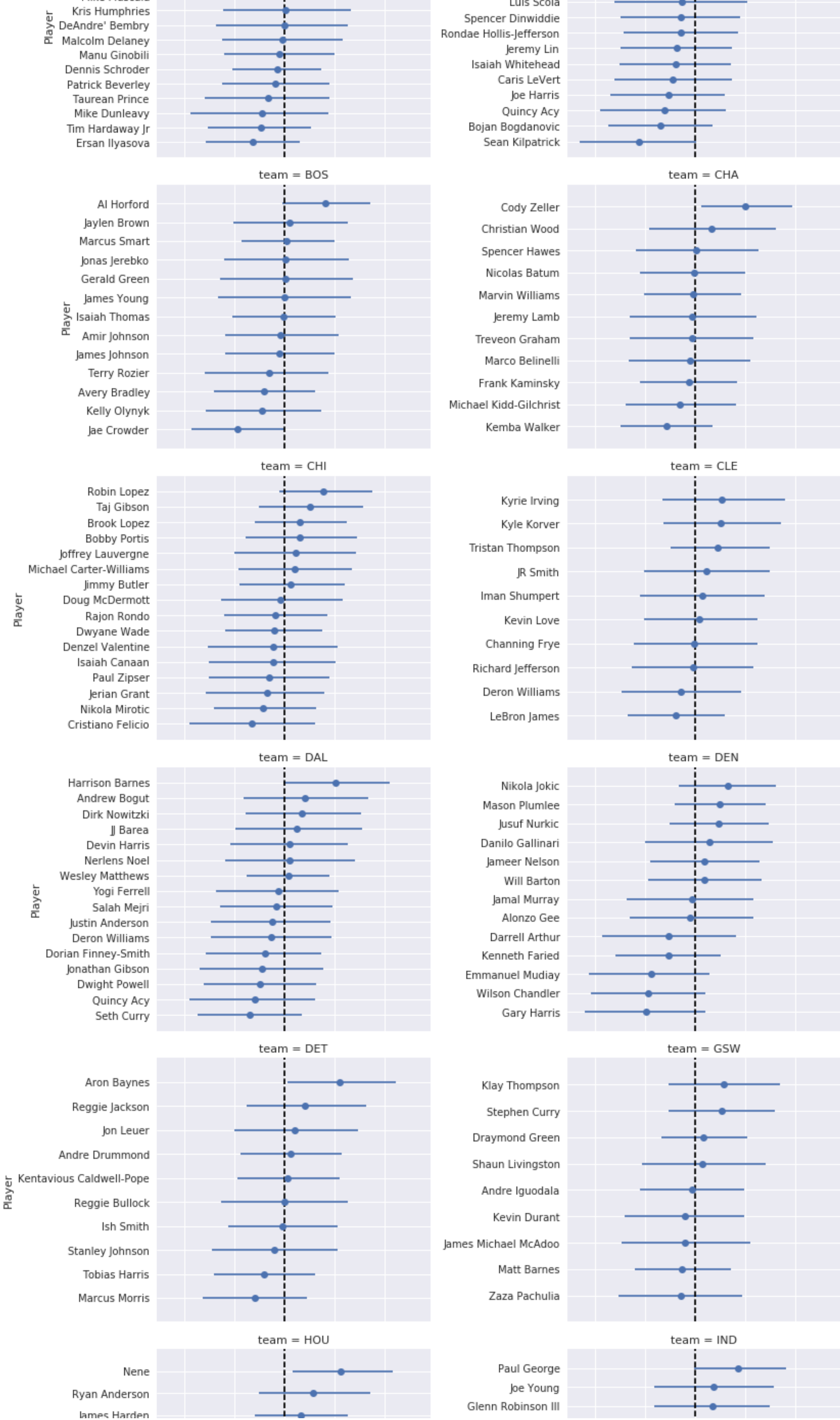

team = CHI
Jimmy Butler
Jerian Grant
Bobby Portis
Nikola Mirotic
Rajon Rondo
Isaiah Canaan
Joffrey Lauvergne
Paul Zipser
Doug McDermott
Cristiano Felicio
Brook Lopez
Taj Gibson
Denzel Valentine
Michael Carter-Williams
Robin Lopez
Dwyane Wade
Player
team = CLE
LeBron James
Kyrie Irving
Kevin Love
Richard Jefferson
Channing Frye
Deron Williams
JR Smith
Tristan Thompson
Kyle Korver
Iman Shumpert
team = DAL
Wesley Matthews
Dorian Finney-Smith
Dirk Nowitzki
Salah Mejri
Seth Curry
Nerlens Noel
Justin Anderson
Quincy Acy
JJ Barea
Jonathan Gibson
Dwight Powell
Andrew Bogut
Deron Williams
Yogi Ferrell
Devin Harris
Harrison Barnes
Player
team = DEN
Emmanuel Mudiay
Mason Plumlee
Will Barton
Wilson Chandler
Kenneth Faried
Danilo Gallinari
Jusuf Nurkic
Jamal Murray
Darrell Arthur
Alonzo Gee
Jameer Nelson
Gary Harris
Nikola Jokic
team = DET
Andre Drummond
Jon Leuer
Reggie Bullock
Stanley Johnson
Reggie Jackson
Aron Baynes
Marcus Morris
Kentavious Caldwell-Pope
Tobias Harris
Ish Smith
Player
team = GSW
Kevin Durant
Draymond Green
Klay Thompson
Zaza Pachulia
James Michael McAdoo
Matt Barnes
Shaun Livingston
Andre Iguodala
Stephen Curry
team = HOU
James Harden
Nene
Patrick Beverley
Trevor Ariza
Montrezl Harrell
Corey Brewer
Eric Gordon
Ryan Anderson
KJ McDaniels
Sam Dekker
Clint Capela
Player
team = IND
Jeff Teague
Monta Ellis
Rodney Stuckey
Myles Turner
Joe Young
Aaron Brooks
Glenn Robinson III
Thaddeus Young
Paul George
Kevin Seraphin
Lavoy Allen
CJ Miles
George Hill
team = LAC
JJ Redick
DeAndre Jordan
Blake Griffin
Chris Paul
Jamal Crawford
Brandon Bass
Luc Mbah a Moute
Paul Pierce
Wesley Johnson
Marreese Speights
Austin Rivers
Player
team = LAL
Brandon Ingram
Lou Williams
Julius Randle
Ivica Zubac
Jordan Clarkson
Tarik Black
Tyler Ennis
David Nwaba
Larry Nance Jr
D'Angelo Russell
Nick Young
Timofey Mozgov
Thomas Robinson

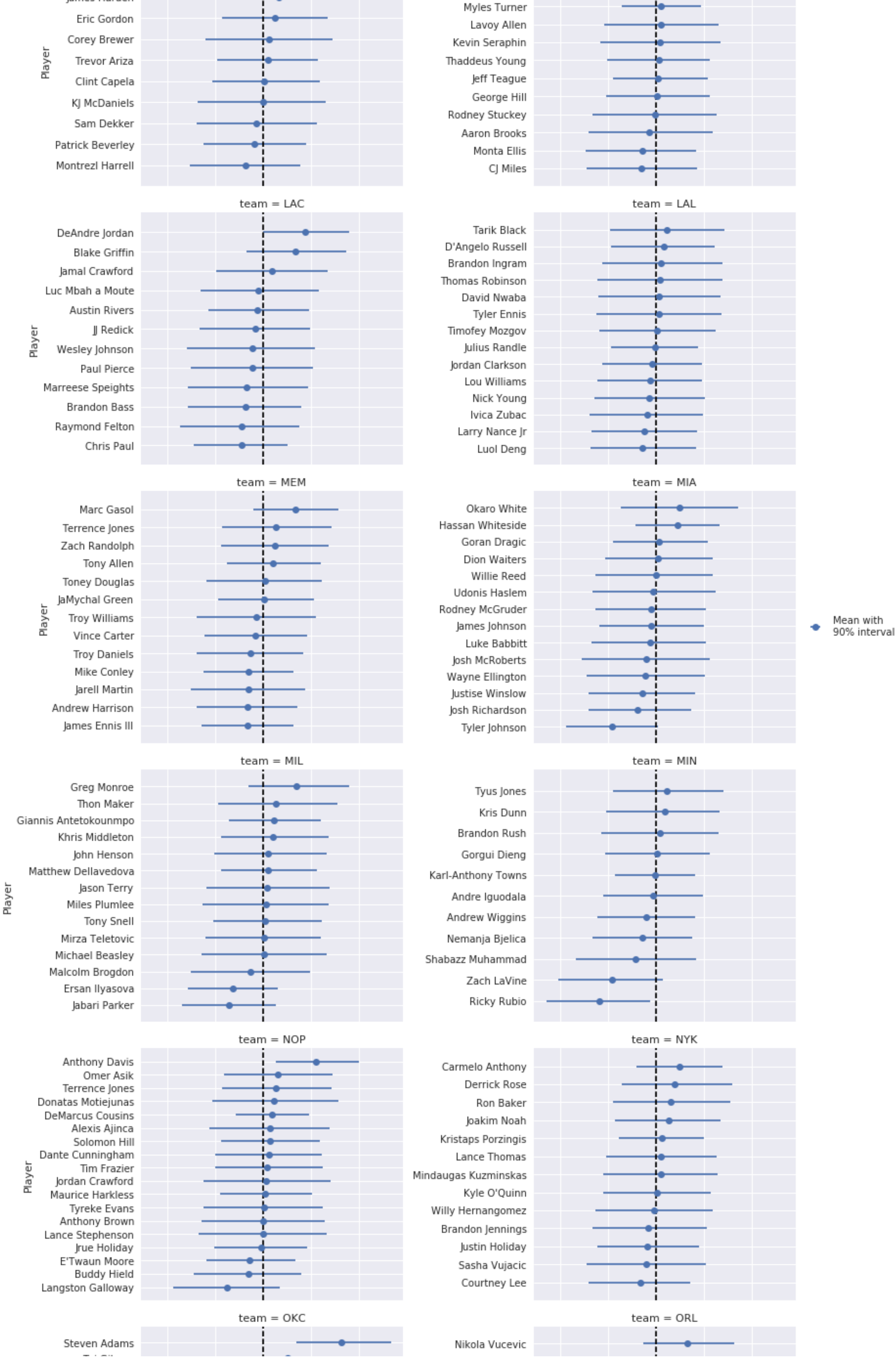


Raymond Felton
Luol Deng
team = MEM
Marc Gasol
Mike Conley
JaMychal Green
Jarell Martin
Tony Allen
James Ennis III
Terrence Jones
Toney Douglas
Troy Williams
Zach Randolph
Troy Daniels
Vince Carter
Andrew Harrison
Player
team = MIA
Hassan Whiteside
Willie Reed
Rodney McGruder
Goran Dragic
Tyler Johnson
Wayne Ellington
Justise Winslow
Luke Babbitt
James Johnson
Udonis Haslem
Okaro White
Josh McRoberts
Josh Richardson
Dion Waiters
μ_θ
Mean with 90% interval
team = MIL
Malcolm Brogdon
Ersan Ilyasova
Greg Monroe
Mirza Teletovic
Jason Terry
Khris Middleton
Miles Plumlee
Jabari Parker
Tony Snell
Thon Maker
Michael Beasley
John Henson
Giannis Antetokounmpo
Matthew Dellavedova
Player
team = MIN
Shabazz Muhammad
Kris Dunn
Tyus Jones
Zach LaVine
Brandon Rush
Gorgui Dieng
Andrew Wiggins
Andre Iguodala
Karl-Anthony Towns
Nemanja Bjelica
Ricky Rubio
team = NOP
Tim Frazier
Jrue Holiday
Tyreke Evans
DeMarcus Cousins
Terrence Jones
Anthony Davis
Dante Cunningham
Solomon Hill
Anthony Brown
Donatas Motiejunas
E'Twaun Moore
Alexis Ajinca
Jordan Crawford
Omer Asik
Lance Stephenson
Maurice Harkless
Buddy Hield
Langston Galloway
Player
team = NYK
Courtney Lee
Kyle O'Quinn
Brandon Jennings
Carmelo Anthony
Sasha Vujacic
Mindaugas Kuzminskas
Ron Baker
Joakim Noah
Justin Holiday
Willy Hernangomez
Kristaps Porzingis
Derrick Rose
Lance Thomas
team = OKC
Russell Westbrook
Kevin Durant
Jerami Grant
Victor Oladipo
Domantas Sabonis
Semaj Christon
Joffrey Lauvergne
Grant, Anthony
Doug McDermott
Taj Gibson
Anthony Morrow
Alex Abrines
Steven Adams
Enes Kanter
Andre Roberson
Player
team = ORL
Evan Fournier
Bismack Biyombo
CJ Watson
Victor Oladipo
Jeff Green
DJ Augustin
Goran Dragic
Serge Ibaka
Nikola Vucevic
Elfrid Payton
Aaron Gordon
Jodie Meeks
Forcier, Chad
Terrence Ross
team = PHI
Joel Embiid
Gerald Henderson
Dario Saric
Ersan Ilyasova
Richaun Holmes
Nerlens Noel
Justin Anderson
Timothe Luwawu-Cabarrot
Hollis Thompson
Jerryd Bayless
Robert Covington
Sergio Rodriguez
Player
team = PHO
PJ Tucker
Alan Williams
Brandon Knight
TJ Warren
Jared Dudley
Marquese Chriss
Dragan Bender
Devin Booker
Alex Len
Tyler Ulis


```
grid = sns.FacetGrid(rasch_df, col='team', col_wrap=2,
                     sharey=False,
                     size=4, aspect=1.5)
grid.map(plot_params_helper,
         'b_mean', 'b_low', 'b_high', 'name',
         league_mean=0.);

grid.set_axis_labels(r"$b$", "Player");

grid.fig.tight_layout();
grid.add_legend();
```




Kris Humphries
DeAndre' Bembry
Malcolm Delaney
Manu Ginobili
Dennis Schroder
Patrick Beverley
Taurean Prince
Mike Dunleavy
Tim Hardaway Jr
Ersan Ilyasova
Player
Luis Scola
Spencer Dinwiddie
Rondae Hollis-Jefferson
Jeremy Lin
Isaiah Whitehead
Caris LeVert
Joe Harris
Quincy Acy
Bojan Bogdanovic
Sean Kilpatrick
team = BOS
Al Horford
Jaylen Brown
Marcus Smart
Jonas Jerebko
Gerald Green
James Young
Isaiah Thomas
Amir Johnson
James Johnson
Terry Rozier
Avery Bradley
Kelly Olynyk
Jae Crowder
Player
team = CHA
Cody Zeller
Christian Wood
Spencer Hawes
Nicolas Batum
Marvin Williams
Jeremy Lamb
Treveon Graham
Marco Belinelli
Frank Kaminsky
Michael Kidd-Gilchrist
Kemba Walker
team = CHI
Robin Lopez
Taj Gibson
Brook Lopez
Bobby Portis
Joffrey Lauvergne
Michael Carter-Williams
Jimmy Butler
Doug McDermott
Rajon Rondo
Dwyane Wade
Denzel Valentine
Isaiah Canaan
Paul Zipser
Jerian Grant
Nikola Mirotic
Cristiano Felicio
Player
team = CLE
Kyrie Irving
Kyle Korver
Tristan Thompson
JR Smith
Iman Shumpert
Kevin Love
Channing Frye
Richard Jefferson
Deron Williams
LeBron James
team = DAL
Harrison Barnes
Andrew Bogut
Dirk Nowitzki
JJ Barea
Devin Harris
Nerlens Noel
Wesley Matthews
Yogi Ferrell
Salah Mejri
Justin Anderson
Deron Williams
Dorian Finney-Smith
Jonathan Gibson
Dwight Powell
Quincy Acy
Seth Curry
Player
team = DEN
Nikola Jokic
Mason Plumlee
Jusuf Nurkic
Danilo Gallinari
Jameer Nelson
Will Barton
Jamal Murray
Alonzo Gee
Darrell Arthur
Kenneth Faried
Emmanuel Mudiay
Wilson Chandler
Gary Harris
team = DET
Aron Baynes
Reggie Jackson
Jon Leuer
Andre Drummond
Kentavious Caldwell-Pope
Reggie Bullock
Ish Smith
Stanley Johnson
Tobias Harris
Marcus Morris
Player
team = GSW
Klay Thompson
Stephen Curry
Draymond Green
Shaun Livingston
Andre Iguodala
Kevin Durant
James Michael McAdoo
Matt Barnes
Zaza Pachulia
team = HOU
Nene
Ryan Anderson
James Harden
team = IND
Paul George
Joe Young
Glenn Robinson III



Eric Gordon
Corey Brewer
Trevor Ariza
Clint Capela
KJ McDaniels
Sam Dekker
Patrick Beverley
Montrezl Harrell
Player
Myles Turner
Lavoy Allen
Kevin Seraphin
Thaddeus Young
Jeff Teague
George Hill
Rodney Stuckey
Aaron Brooks
Monta Ellis
CJ Miles
team = LAC
DeAndre Jordan
Blake Griffin
Jamal Crawford
Luc Mbah a Moute
Austin Rivers
JJ Redick
Wesley Johnson
Paul Pierce
Marreese Speights
Brandon Bass
Raymond Felton
Chris Paul
Player
team = LAL
Tarik Black
D'Angelo Russell
Brandon Ingram
Thomas Robinson
David Nwaba
Tyler Ennis
Timofey Mozgov
Julius Randle
Jordan Clarkson
Lou Williams
Nick Young
Ivica Zubac
Larry Nance Jr
Luol Deng
team = MEM
Marc Gasol
Terrence Jones
Zach Randolph
Tony Allen
Toney Douglas
JaMychal Green
Troy Williams
Vince Carter
Troy Daniels
Mike Conley
Jarell Martin
Andrew Harrison
James Ennis III
Player
team = MIA
Okaro White
Hassan Whiteside
Goran Dragic
Dion Waiters
Willie Reed
Udonis Haslem
Rodney McGruder
James Johnson
Luke Babbitt
Josh McRoberts
Wayne Ellington
Justise Winslow
Josh Richardson
Tyler Johnson
Mean with
90% interval
team = MIL
Greg Monroe
Thon Maker
Giannis Antetokounmpo
Khris Middleton
John Henson
Matthew Dellavedova
Jason Terry
Miles Plumlee
Tony Snell
Mirza Teletovic
Michael Beasley
Malcolm Brogdon
Ersan Ilyasova
Jabari Parker
Player
team = MIN
Tyus Jones
Kris Dunn
Brandon Rush
Gorgui Dieng
Karl-Anthony Towns
Andre Iguodala
Andrew Wiggins
Nemanja Bjelica
Shabazz Muhammad
Zach LaVine
Ricky Rubio
team = NOP
Anthony Davis
Omer Asik
Terrence Jones
Donatas Motiejunas
DeMarcus Cousins
Alexis Ajinca
Solomon Hill
Dante Cunningham
Tim Frazier
Jordan Crawford
Maurice Harkless
Tyreke Evans
Anthony Brown
Lance Stephenson
Jrue Holiday
E'Twaun Moore
Buddy Hield
Langston Galloway
Player
team = NYK
Carmelo Anthony
Derrick Rose
Ron Baker
Joakim Noah
Kristaps Porzingis
Lance Thomas
Mindaugas Kuzminskas
Kyle O'Quinn
Willy Hernangomez
Brandon Jennings
Justin Holiday
Sasha Vujacic
Courtney Lee
team = OKC
Steven Adams
team = ORL
Nikola Vucevic

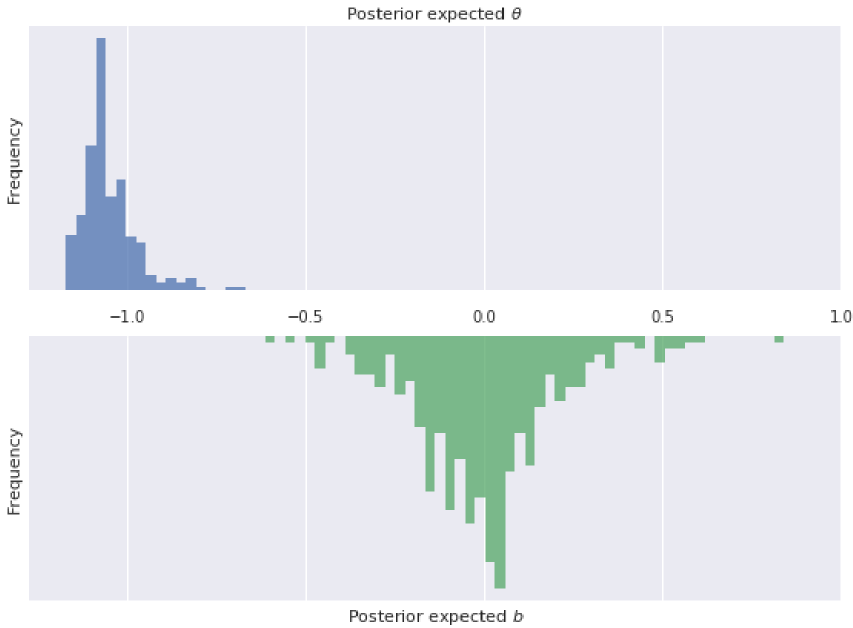

Taj Gibson
Joffrey Lauvergne
Grant, Anthony
Domantas Sabonis
Semaj Christon
Enes Kanter
Russell Westbrook
Anthony Morrow
Andre Roberson
Doug McDermott
Victor Oladipo
Kevin Durant
Alex Abrines
Jerami Grant
Player
Terrence Ross
Evan Fournier
Forcier, Chad
Goran Dragic
Bismack Biyombo
CJ Watson
DJ Augustin
Elfrid Payton
Victor Oladipo
Aaron Gordon
Serge Ibaka
Jodie Meeks
Jeff Green
team = PHI
Jahlil Okafor
Dario Saric
Nerlens Noel
Timothe Luwawu-Cabarrot
Sergio Rodriguez
Jerryd Bayless
TJ McConnell
Richaun Holmes
Justin Anderson
Hollis Thompson
Nik Stauskas
Joel Embiid
Gerald Henderson
Robert Covington
Ersan Ilyasova
Player
team = PHO
Tyson Chandler
Brandon Knight
Marquese Chriss
Alex Len
Eric Bledsoe
Dragan Bender
Leandro Barbosa
Alan Williams
PJ Tucker
Tyler Ulis
Devin Booker
TJ Warren
Jared Dudley
team = POR
Al-Farouq Aminu
Mason Plumlee
Jusuf Nurkic
Evan Turner
Meyers Leonard
Shabazz Napier
Ed Davis
Maurice Harkless
Pat Connaughton
Allen Crabbe
CJ McCollum
Damian Lillard
Player
team = SAC
Willie Cauley-Stein
Anthony Tolliver
Kosta Koufos
DeMarcus Cousins
Ben McLemore
Arron Afflalo
Skal Labissiere
Tyreke Evans
Garrett Temple
Omri Casspi
Ty Lawson
Matt Barnes
Justise Winslow
Buddy Hield
Rudy Gay
Darren Collison
Langston Galloway
team = SAS
LaMarcus Aldridge
Kawhi Leonard
Paul Millsap
David Lee
Danny Green
Dewayne Dedmon
Kyle Anderson
Dejounte Murray
Davis Bertans
Pau Gasol
Jonathon Simmons
Tony Parker
Manu Ginobili
Patty Mills
Player
team = TOR
Patrick Patterson
Terrence Ross
Jonas Valanciunas
DeMarre Carroll
Cory Joseph
Delon Wright
Fred VanVleet
DeMar DeRozan
Lucas Nogueira
Jakob Poeltl
Norman Powell
Serge Ibaka
PJ Tucker
Kyle Lowry
team = UTA
Rudy Gobert
Derrick Favors
Joe Johnson
Trey Lyles
George Hill
Rodney Hood
Gordon Hayward
James Johnson
Alec Burks
Jeff Withey
Raul Neto
Joe Ingles
Shelvin Mack
Boris Diaw
Tyler Johnson
Player
team = WAS
Marcin Gortat
John Wall
Markieff Morris
Otto Porter Jr
Bradley Beal
Ian Mahinmi
Kelly Oubre Jr
Marcus Thornton
Tomas Satoransky
Marcus Morris
Bojan Bogdanovic
−1.0
−0.5
0.0
0.5
1.0

b b

Though these plots are voluminuous and therefore difficult to interpret precisely, a few trends are evident. The first is that there is more variation in the committing skill (b_j) than in disadvantaged skill (θ_i). This difference is confirmed in the following histograms of the posterior expected values of θ_i and b_j.

```
def plot_latent_distributions(θ, b):
    fig, (θ_ax, b_ax) = plt.subplots(nrows=2, sharex=True, figsize=(8, 6))

    bins = np.linspace(0.9 * min(θ.min(), b.min()),
                       1.1 * max(θ.max(), b.max()),
                       75)

    θ_ax.hist(θ, bins=bins,
              alpha=0.75)

    θ_ax.xaxis.set_label_position('top')
    θ_ax.set_xlabel(r"Posterior expected $\theta$")

    θ_ax.set_yticks([])
    θ_ax.set_ylabel("Frequency")

    b_ax.hist(b, bins=bins,
              color=green, alpha=0.75)

    b_ax.xaxis.tick_top()
    b_ax.set_xlabel(r"Posterior expected $b$")

    b_ax.set_yticks([])
    b_ax.invert_yaxis()
    b_ax.set_ylabel("Frequency")

    fig.tight_layout()
```

```
plot_latent_distributions(rasch_df.θ_mean, rasch_df.b_mean)
```

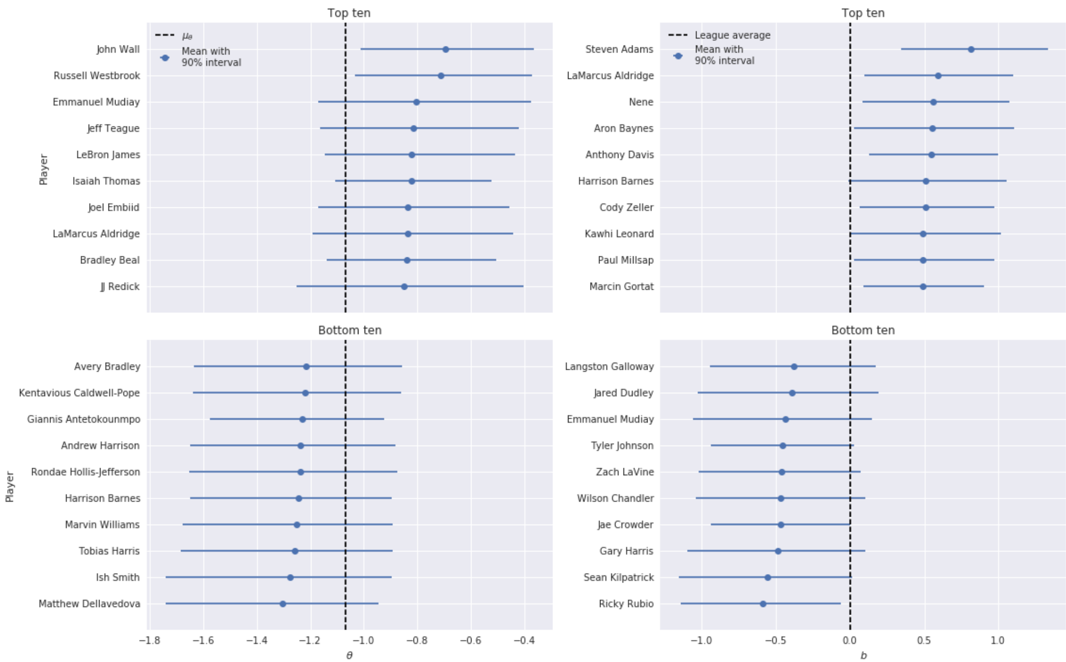


The following plots show the top and bottom ten players in terms of both θ_i and b_j.

```python
def top_10_plot(trace_df, μ_θ=0):
    fig = plt.figure(figsize=(16, 10))
    θ_top_ax = fig.add_subplot(221)
    b_top_ax = fig.add_subplot(222)
    θ_bottom_ax = fig.add_subplot(223, sharex=θ_top_ax)
    b_bottom_ax = fig.add_subplot(224, sharex=b_top_ax)

    # necessary for players that have been on more than one team
    trace_df = trace_df.drop_duplicates(['player_id'])

    by_θ = trace_df.sort_values('θ_mean')
    θ_top = by_θ.iloc[-10:]
    θ_bottom = by_θ.iloc[:10]

    plot_params(θ_top.θ_mean.values, θ_top[['θ_low', 'θ_high']].values.T,
                θ_top.name.values,
                ax=θ_top_ax)
    θ_top_ax.vlines(μ_θ, -1, 10,
                    'k', '--',
                    label=(r"$\mu_{\theta}$" if μ_θ != 0 else "League average"))

    plt.setp(θ_top_ax.get_xticklabels(), visible=False)

    θ_top_ax.set_ylabel("Player")

    θ_top_ax.legend(loc=2)
    θ_top_ax.set_title("Top ten")

    plot_params(θ_bottom.θ_mean.values, θ_bottom[['θ_low', 'θ_high']].values.T,
                θ_bottom.name.values,
                ax=θ_bottom_ax)
    θ_bottom_ax.vlines(μ_θ, -1, 10,
                       'k', '--',
                       label=(r"$\mu_{\theta}$" if μ_θ != 0 else "League average"))

    θ_bottom_ax.set_xlabel(r"$\theta$")

    θ_bottom_ax.set_ylabel("Player")

    θ_bottom_ax.set_title("Bottom ten")

    by_b = trace_df.sort_values('b_mean')
    b_top = by_b.iloc[-10:]
    b_bottom = by_b.iloc[:10]

    plot_params(b_top.b_mean.values, b_top[['b_low', 'b_high']].values.T,
                b_top.name.values,
                ax=b_top_ax)
    b_top_ax.vlines(0, -1, 10,
                    'k', '--',
                    label="League average");

    plt.setp(b_top_ax.get_xticklabels(), visible=False)

    b_top_ax.legend(loc=2)
    b_top_ax.set_title("Top ten")
```

```
    b_bottom_player_id = b.argsort()[:10]

    plot_params(b_bottom.b_mean.values, b_bottom[['b_low', 'b_high']].values.T,
                b_bottom.name.values,
                ax=b_bottom_ax)
    b_bottom_ax.vlines(0, -1, 10,
                       'k', '--')

    b_bottom_ax.set_xlabel(r"$b$")

    b_bottom_ax.set_title("Bottom ten")

    fig.tight_layout()
```

```
top_10_plot(rasch_df, μ_θ=μ_θ_mean)
```



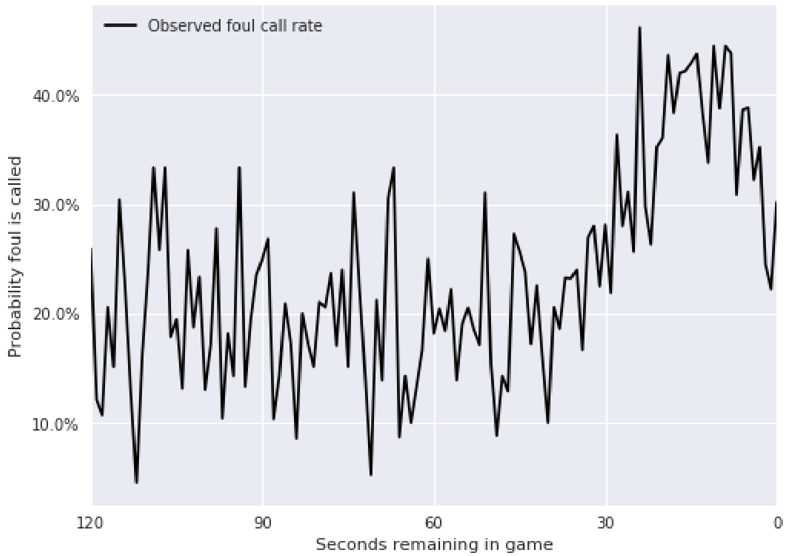
We focus first on θ_i. Interestingly, the top-ten players for the Rasch model contains many more top-tier stars than the logistic-normal model, including John Wall, Russell Westbrook, and LeBron James. Turning to b, it is interesting that the while the top and bottom ten players contain many recognizable names (LaMarcus Aldridge, Harrison Barnes, Kawhi Leonard, and Ricky Rubio) the only truly top-tier player present is Anthony Davis.

Time remaining model

As basketball fans know, there are many factors other than the players involved that influence foul calls. Very often, sufficiently close NBA games end with intentional fouls, as the losing team attempts to stop the clock and force another offensive possesion. Therefore, we expect to see in increase in the foul call probability as the game nears its conclusion.

```
n_sec = 121
sec = (df.seconds_left
         .round(0)
         .values
         .astype(np.int64))
```

```
fig, ax = plt.subplots(figsize=(8, 6))

(df.groupby(sec)
   .foul_called
   .mean()
   .plot(c='k', label="Observed foul call rate", ax=ax));

ax.set_xticks(np.linspace(0, 120, 5));
ax.invert_xaxis();
ax.set_xlabel("Seconds remaining in game");

ax.yaxis.set_major_formatter(pct_formatter);
ax.set_ylabel("Probability foul is called");

ax.legend(loc=2);
```



The plot above confirms this expectation, which we can use to improve our latent skill model. If $t \in \{0, 1, \ldots, 120\}$ is the number of seconds remaining in the game, we model the latent contribution of t to the logodds that a foul is called with a Gaussian random walk (https://en.wikipedia.org/wiki/Random_walk#Gaussian_random_walk),

$$
\begin{aligned}
\lambda_0 &\sim N(0, 100) \\
\lambda_t &\sim N(\lambda_{t-1}, \tau_\lambda^{-1}) \\
\tau_\lambda &\sim \text{Exp}(10^{-4}).
\end{aligned}
$$

This prior allows us to flexibly model the shape of the curve shown above. If $t(k)$ is the number of seconds remaining during the k-th play, we incorporate $\lambda_{t(k)}$ into our model with

$$\eta_k = \lambda_{t(k)} + \theta_{i(k)} - b_{j(k)}.$$

This model is not identified until we constrain the mean of θ to be zero, for reasons similar to those discussed above for the Rasch model.

```
with pm.Model() as time_model:
    τ_λ = pm.Exponential('τ_λ', 1e-4)
    λ = pm.GaussianRandomWalk('λ', tau=τ_λ,
                              init=pm.Normal.dist(0., 10.),
                              shape=n_sec)

    Δ_θ = pm.Normal('Δ_θ', 0., 1., shape=n_players)
    σ_θ = pm.HalfCauchy('σ_θ', 2.5)
    θ = pm.Deterministic('θ', Δ_θ * σ_θ)

    Δ_b = pm.Normal('Δ_b', 0., 1., shape=n_players)
    σ_b = pm.HalfCauchy('σ_b', 2.5)
    b = pm.Deterministic('b', Δ_b * σ_b)

    η = λ[sec] + θ[disadvantaged_id] - b[committing_id]
    p = pm.Deterministic('p', pm.math.sigmoid(η))

    y = pm.Bernoulli('y_obs', p, observed=foul_called)
```

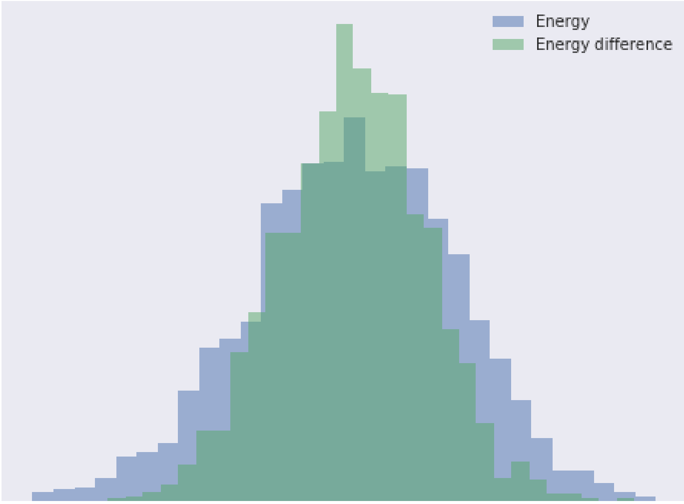
We now sample from the model.

```
time_trace = sample(time_model, N_TUNE, N_SAMPLES, SEED)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using advi...
Average ELBO = -1.0533e+05:  15%|█         | 30300/200000 [00:31<02:52, 982.03it/s] Median ELB
O converged.
Finished [100%]: Average ELBO = -3,104.5

100%|██████████| 4000/4000 [03:10<00:00, 20.99it/s]
```

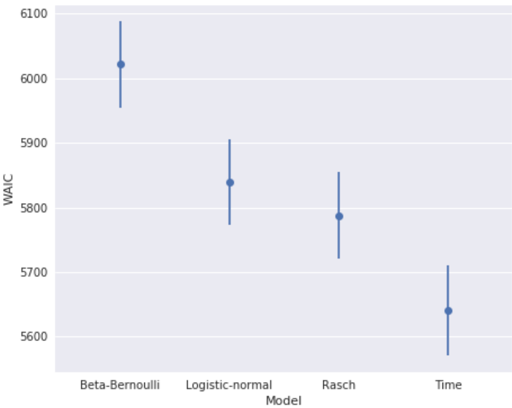
The energy plot for this model is worse than the previous ones, but not too bad.

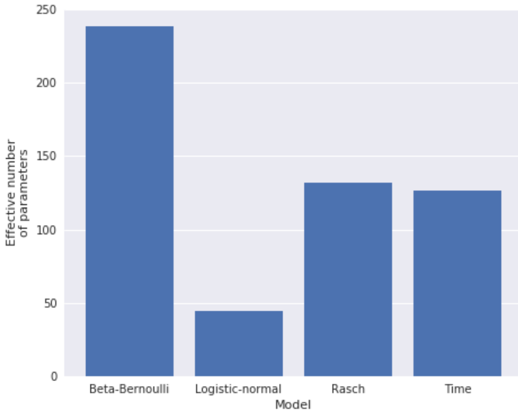
```
energy_plot(time_trace)
```




```
waic_df = waic_df.append(get_waic_df(time_model, time_trace, "Time"))
```

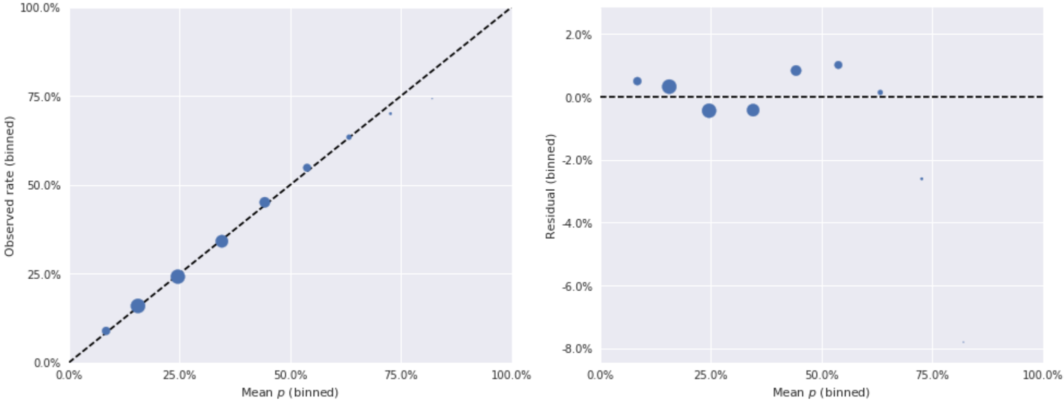
```
waic_plot(waic_df)
```





We see that the time remaining model represents an appreciable improvement over the Rasch model in terms of WAIC.

```
bin_obs, bin_p, bin_counts = binned_residuals(foul_called, time_trace['p'])

binned_residual_plot(bin_obs, bin_p, bin_counts)
```

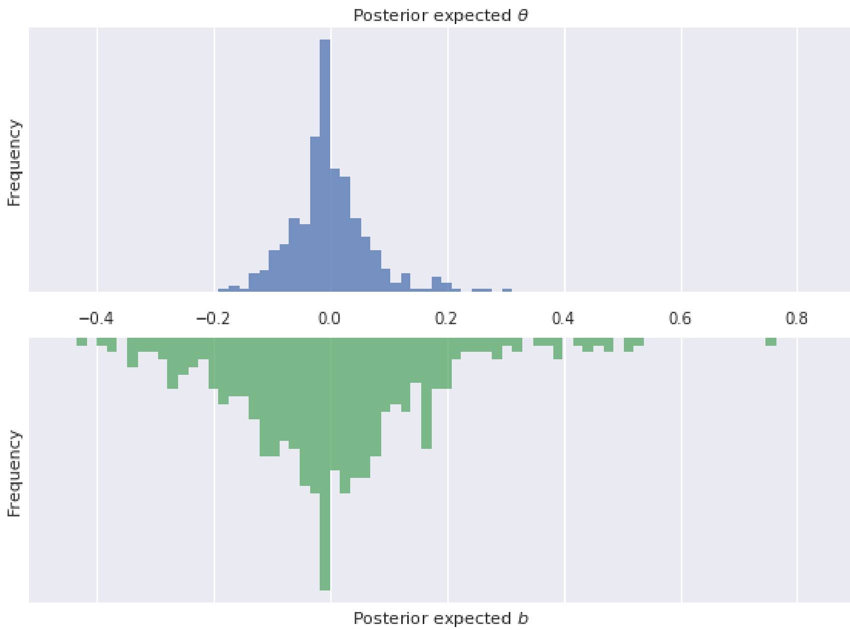


The binned residuals for this model also look quite good, with very few samples appreciably exceeding a 1% difference.

We now compare the distribtuions of θ_i and b_j for this model with those for the Rasch model.

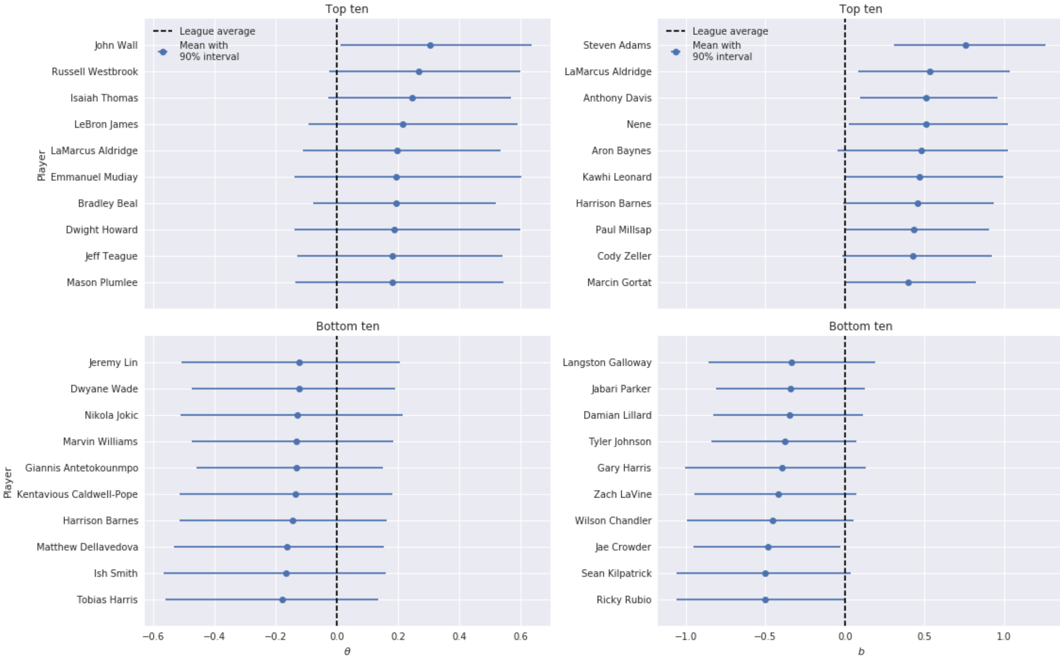
```
time_df = to_param_df(team_player_map, time_trace, ['θ', 'b'])
```

```
plot_latent_distributions(time_df.θ_mean, time_df.b_mean)
```



The effect of constraining the mean of θ to be zero is immediately apparent. Also, the variation in θ remains lower than the variation than b in this model. We also see that the top- and bottom-ten players by θ- and b-value remain largely unchanged from the Rasch model.

```
top_10_plot(time_df)
```

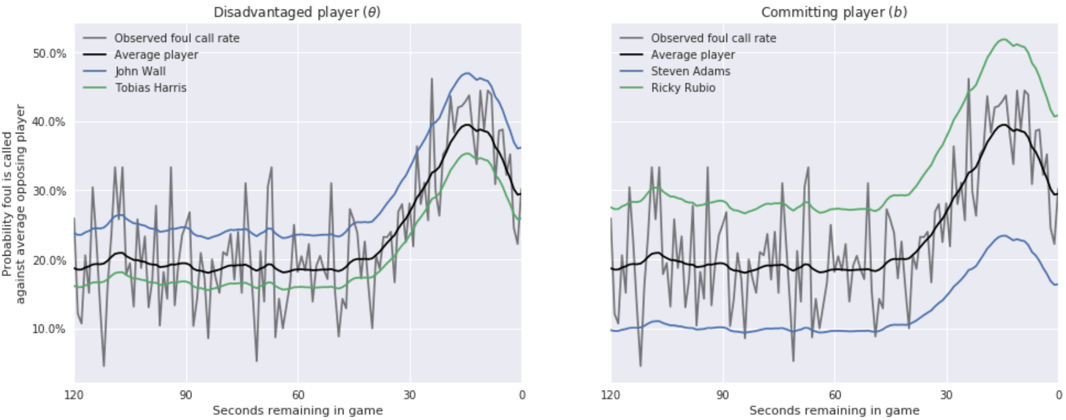


Basketball fans may find it amusing that under this model, Dwight Howard has joined the top-ten in terms of θ and Ricky Rubio is no longer the worst player in terms of b.

While this model has not done much to change the rank-ordering of the most- and least-skilled players, it does enable us to plot per-player foul probabilities over time, as below.

```
fig, (θ_ax, b_ax) = plt.subplots(ncols=2, sharex=True, sharey=True, figsize=(16, 6))

(df.groupby(sec)
   .foul_called
   .mean()
   .plot(c='k', alpha=0.5,
         label="Observed foul call rate",
         ax=θ_ax));

plot_sec = np.arange(n_sec)

θ_ax.plot(plot_sec, expit(time_trace['λ'].mean(axis=0)),
          c='k',
          label="Average player");

θ_best_id = time_df.ix[time_df.θ_mean.idxmax()].player_id
θ_ax.plot(plot_sec, expit(time_trace['θ'][:, θ_best_id].mean(axis=0) \
                          + time_trace['λ'].mean(axis=0)),
          label=player_map[θ_best_id]);

θ_worst_id = time_df.ix[time_df.θ_mean.idxmin()].player_id
θ_ax.plot(plot_sec, expit(time_trace['θ'][:, θ_worst_id].mean(axis=0) \
                          + time_trace['λ'].mean(axis=0)),
          label=player_map[θ_worst_id]);


θ_ax.set_xticks(np.linspace(0, 120, 5));
θ_ax.invert_xaxis();
θ_ax.set_xlabel("Seconds remaining in game");

θ_ax.yaxis.set_major_formatter(pct_formatter);
θ_ax.set_ylabel("Probability foul is called\nagainst average opposing player");

θ_ax.legend(loc=2);
θ_ax.set_title(r"Disadvantaged player ($\theta$)");

(df.groupby(sec)
   .foul_called
   .mean()
   .plot(c='k', alpha=0.5,
         label="Observed foul call rate",
         ax=b_ax));

plot_sec = np.arange(n_sec)

b_ax.plot(plot_sec, expit(time_trace['λ'].mean(axis=0)),
          c='k',
          label="Average player");

b_best_id = time_df.ix[time_df.b_mean.idxmax()].player_id
b_ax.plot(plot_sec, expit(-time_trace['b'][:, b_best_id].mean(axis=0) \
                          + time_trace['λ'].mean(axis=0)),
          label=player_map[b_best_id]);

b_worst_id = time_df.ix[time_df.b_mean.idxmin()].player_id
b_ax.plot(plot_sec, expit(-time_trace['b'][:, b_worst_id].mean(axis=0) \
```

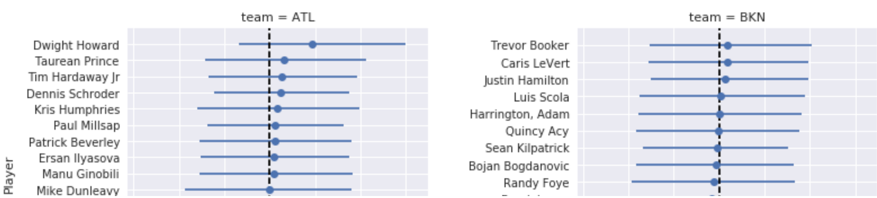
```
                            + time_trace['λ'].mean(axis=0)),
            label=player_map[b_worst_id]);


b_ax.set_xticks(np.linspace(0, 120, 5));
b_ax.invert_xaxis();
b_ax.set_xlabel("Seconds remaining in game");

b_ax.legend(loc=2);
b_ax.set_title(r"Committing player ($b$)");
```



Here we have plotted the probability of a foul call while being opposed by an average player (for both θ and b), along with the probability curves for the players with the highest and lowest θ and b, respectively. While these plots are quite interesting, one weakness of our model is that the difference between each player's curve and the league average is constant over time. It would be an interesting and useful to extend this model to allow player offsets to vary over time. Additonally, it would be interesting to understand the influence of the score on the foul-called rate as the game nears its end. It seems quite likely that the winning team is much less likely to commit fouls while the losing team is much more likely to to commit intentional fouls in close games.

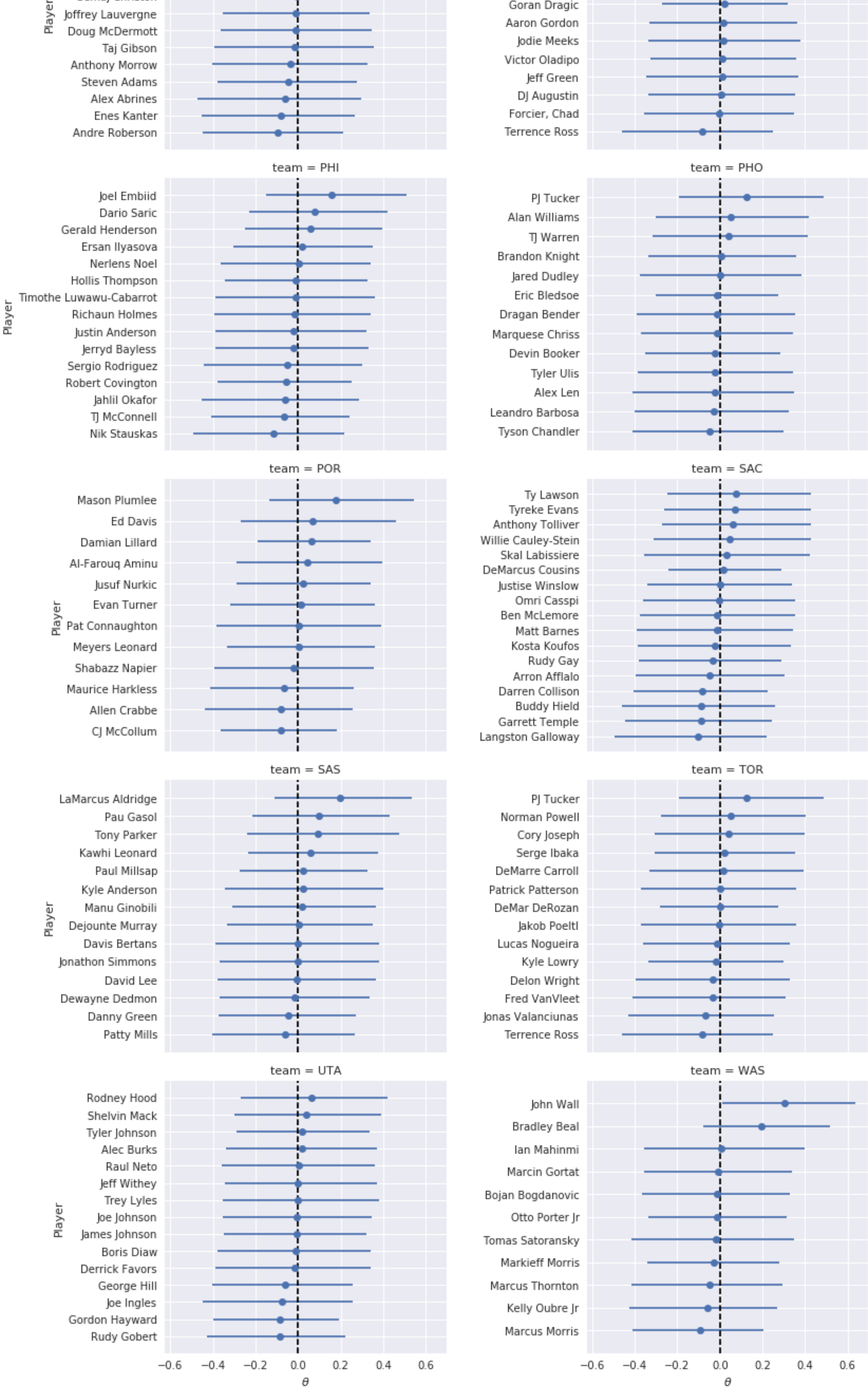
We now plot the per-player values of θ_i and b_j under this model.

```
grid = sns.FacetGrid(time_df, col='team', col_wrap=2,
                     sharey=False,
                     size=4, aspect=1.5)
grid.map(plot_params_helper,
         'θ_mean', 'θ_low', 'θ_high', 'name',
         league_mean=0.,
         league_mean_name="League average");

grid.set_axis_labels(r"$\theta$", "Player");

grid.fig.tight_layout();
grid.add_legend();
```



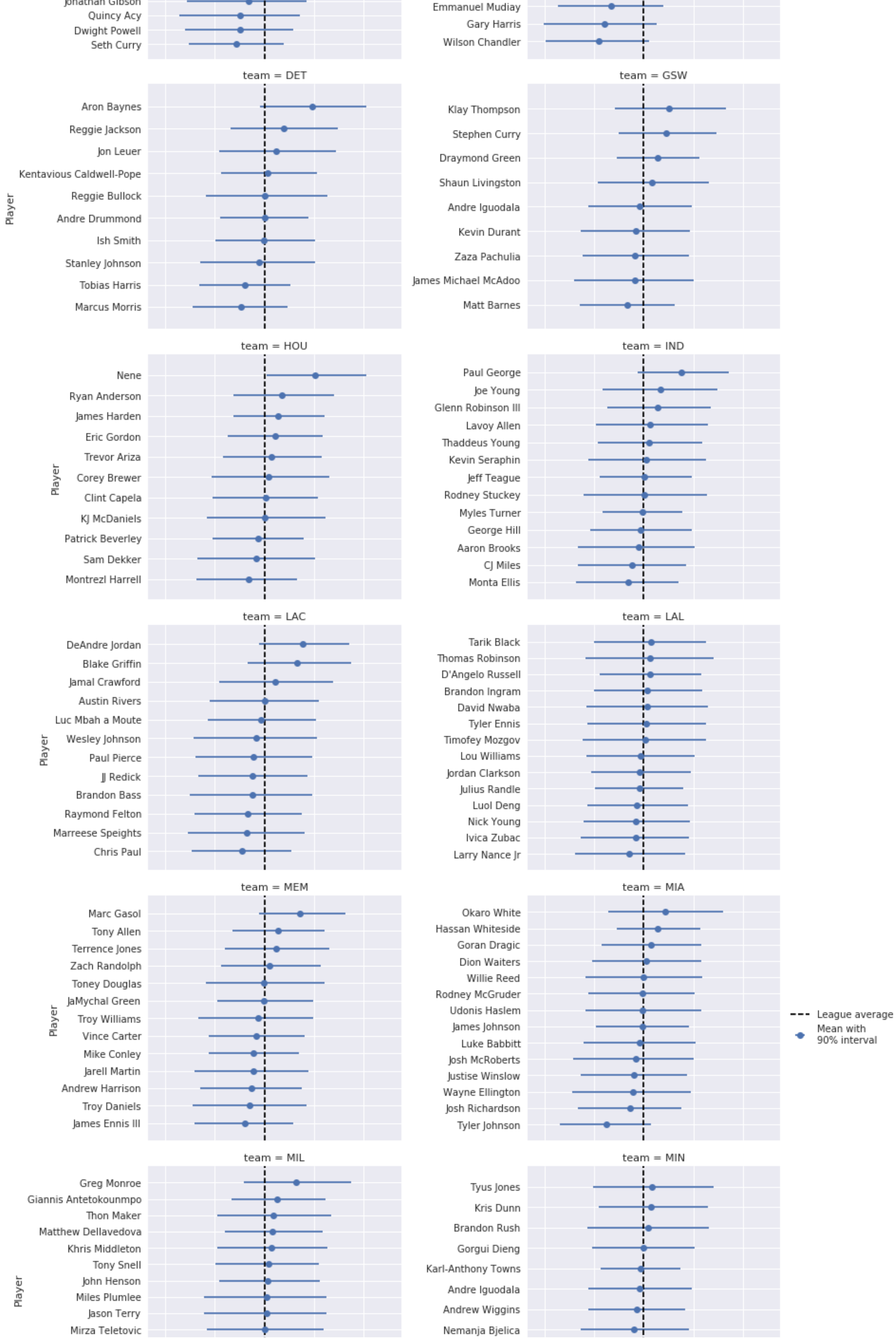

Thabo Sefolosha
DeAndre' Bembry
Mike Muscala
Malcolm Delaney
Kent Bazemore
Derrick Rose
Kyle Korver
Brook Lopez
Spencer Dinwiddie
Yogi Ferrell
Joe Harris
Isaiah Whitehead
Rondae Hollis-Jefferson
Jeremy Lin
team = BOS
Isaiah Thomas
Jaylen Brown
Jae Crowder
Amir Johnson
Kelly Olynyk
Marcus Smart
Jonas Jerebko
James Johnson
James Young
Gerald Green
Terry Rozier
Al Horford
Avery Bradley
Player
team = CHA
Kemba Walker
Cody Zeller
Jeremy Lamb
Christian Wood
Frank Kaminsky
Spencer Hawes
Treveon Graham
Michael Kidd-Gilchrist
Nicolas Batum
Marco Belinelli
Marvin Williams
team = CHI
Jimmy Butler
Jerian Grant
Nikola Mirotic
Bobby Portis
Rajon Rondo
Isaiah Canaan
Joffrey Lauvergne
Doug McDermott
Taj Gibson
Paul Zipser
Denzel Valentine
Cristiano Felicio
Brook Lopez
Michael Carter-Williams
Robin Lopez
Dwyane Wade
Player
team = CLE
LeBron James
Kevin Love
Kyrie Irving
Richard Jefferson
Channing Frye
Deron Williams
JR Smith
Tristan Thompson
Iman Shumpert
Kyle Korver
team = DAL
Wesley Matthews
Dorian Finney-Smith
Dirk Nowitzki
Salah Mejri
Seth Curry
Nerlens Noel
Quincy Acy
JJ Barea
Justin Anderson
Jonathan Gibson
Andrew Bogut
Deron Williams
Dwight Powell
Yogi Ferrell
Devin Harris
Harrison Barnes
Player
team = DEN
Emmanuel Mudiay
Mason Plumlee
Will Barton
Kenneth Faried
Wilson Chandler
Danilo Gallinari
Jusuf Nurkic
Jamal Murray
Darrell Arthur
Alonzo Gee
Jameer Nelson
Gary Harris
Nikola Jokic
team = DET
Andre Drummond
Jon Leuer
Reggie Bullock
Stanley Johnson
Reggie Jackson
Aron Baynes
Marcus Morris
Kentavious Caldwell-Pope
Ish Smith
Tobias Harris
Player
team = GSW
Kevin Durant
Draymond Green
Klay Thompson
Zaza Pachulia
James Michael McAdoo
Shaun Livingston
Matt Barnes
Andre Iguodala
Stephen Curry
team = HOU
James Harden
Nene
Patrick Beverley
Trevor Ariza
Montrezl Harrell
Eric Gordon
team = IND
Jeff Teague
Monta Ellis
Rodney Stuckey
Myles Turner
Paul George
Joe Young
Aaron Brooks


Corey Brewer
KJ McDaniels
Ryan Anderson
Clint Capela
Sam Dekker
Thaddeus Young
Kevin Seraphin
Glenn Robinson III
Lavoy Allen
CJ Miles
George Hill
team = LAC
JJ Redick
DeAndre Jordan
Blake Griffin
Chris Paul
Austin Rivers
Jamal Crawford
Marreese Speights
Paul Pierce
Wesley Johnson
Luc Mbah a Moute
Raymond Felton
Brandon Bass
Player
team = LAL
Brandon Ingram
Jordan Clarkson
Ivica Zubac
Julius Randle
Lou Williams
Tarik Black
Tyler Ennis
David Nwaba
Timofey Mozgov
Nick Young
Thomas Robinson
Larry Nance Jr
D'Angelo Russell
Luol Deng
team = MEM
Marc Gasol
Mike Conley
JaMychal Green
Tony Allen
James Ennis III
Jarell Martin
Terrence Jones
Toney Douglas
Zach Randolph
Troy Williams
Troy Daniels
Andrew Harrison
Vince Carter
Player
team = MIA
Hassan Whiteside
Willie Reed
Rodney McGruder
Goran Dragic
Tyler Johnson
Okaro White
Luke Babbitt
Justise Winslow
Wayne Ellington
James Johnson
Udonis Haslem
Josh McRoberts
Josh Richardson
Dion Waiters
League average
Mean with
90% interval
team = MIL
Malcolm Brogdon
Ersan Ilyasova
Mirza Teletovic
Greg Monroe
Jason Terry
Khris Middleton
Miles Plumlee
Thon Maker
Michael Beasley
Tony Snell
Jabari Parker
John Henson
Giannis Antetokounmpo
Matthew Dellavedova
Player
team = MIN
Shabazz Muhammad
Kris Dunn
Tyus Jones
Gorgui Dieng
Brandon Rush
Zach LaVine
Karl-Anthony Towns
Andrew Wiggins
Nemanja Bjelica
Andre Iguodala
Ricky Rubio
team = NOP
Tim Frazier
Jrue Holiday
Tyreke Evans
Anthony Davis
Terrence Jones
Dante Cunningham
Solomon Hill
DeMarcus Cousins
Anthony Brown
Donatas Motiejunas
Alexis Ajinca
Omer Asik
Jordan Crawford
E'Twaun Moore
Lance Stephenson
Maurice Harkless
Buddy Hield
Langston Galloway
Player
team = NYK
Courtney Lee
Kyle O'Quinn
Brandon Jennings
Mindaugas Kuzminskas
Sasha Vujacic
Ron Baker
Justin Holiday
Joakim Noah
Carmelo Anthony
Kristaps Porzingis
Willy Hernangomez
Derrick Rose
Lance Thomas
team = OKC
Russell Westbrook
Kevin Durant
Jerami Grant
Victor Oladipo
Domantas Sabonis
Grant, Anthony
Semaj Christon
team = ORL
Evan Fournier
Bismack Biyombo
CJ Watson
Elfrid Payton
Serge Ibaka
Nikola Vucevic



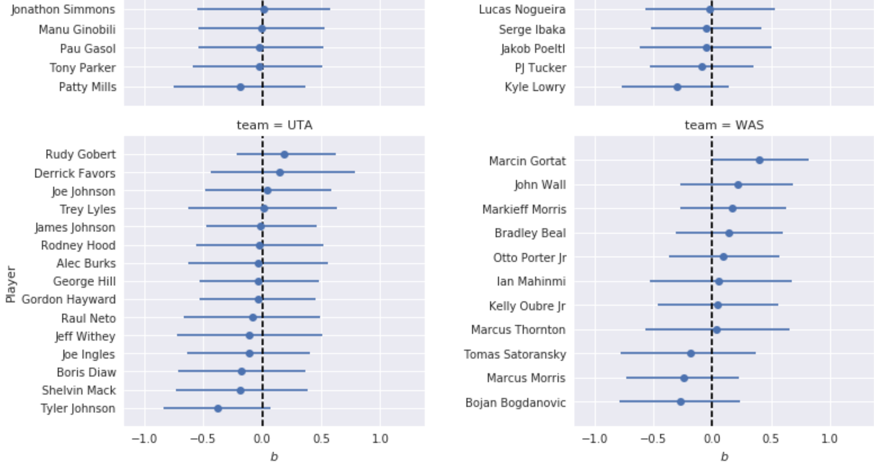
Player
Joffrey Lauvergne
Doug McDermott
Taj Gibson
Anthony Morrow
Steven Adams
Alex Abrines
Enes Kanter
Andre Roberson
Goran Dragic
Aaron Gordon
Jodie Meeks
Victor Oladipo
Jeff Green
DJ Augustin
Forcier, Chad
Terrence Ross
team = PHI
Joel Embiid
Dario Saric
Gerald Henderson
Ersan Ilyasova
Nerlens Noel
Hollis Thompson
Timothe Luwawu-Cabarrot
Richaun Holmes
Justin Anderson
Jerryd Bayless
Sergio Rodriguez
Robert Covington
Jahlil Okafor
TJ McConnell
Nik Stauskas
team = PHO
PJ Tucker
Alan Williams
TJ Warren
Brandon Knight
Jared Dudley
Eric Bledsoe
Dragan Bender
Marquese Chriss
Devin Booker
Tyler Ulis
Alex Len
Leandro Barbosa
Tyson Chandler
team = POR
Mason Plumlee
Ed Davis
Damian Lillard
Al-Farouq Aminu
Jusuf Nurkic
Evan Turner
Pat Connaughton
Meyers Leonard
Shabazz Napier
Maurice Harkless
Allen Crabbe
CJ McCollum
team = SAC
Ty Lawson
Tyreke Evans
Anthony Tolliver
Willie Cauley-Stein
Skal Labissiere
DeMarcus Cousins
Justise Winslow
Omri Casspi
Ben McLemore
Matt Barnes
Kosta Koufos
Rudy Gay
Arron Afflalo
Darren Collison
Buddy Hield
Garrett Temple
Langston Galloway
team = SAS
LaMarcus Aldridge
Pau Gasol
Tony Parker
Kawhi Leonard
Paul Millsap
Kyle Anderson
Manu Ginobili
Dejounte Murray
Davis Bertans
Jonathon Simmons
David Lee
Dewayne Dedmon
Danny Green
Patty Mills
team = TOR
PJ Tucker
Norman Powell
Cory Joseph
Serge Ibaka
DeMarre Carroll
Patrick Patterson
DeMar DeRozan
Jakob Poeltl
Lucas Nogueira
Kyle Lowry
Delon Wright
Fred VanVleet
Jonas Valanciunas
Terrence Ross
team = UTA
Rodney Hood
Shelvin Mack
Tyler Johnson
Alec Burks
Raul Neto
Jeff Withey
Trey Lyles
Joe Johnson
James Johnson
Boris Diaw
Derrick Favors
George Hill
Joe Ingles
Gordon Hayward
Rudy Gobert
team = WAS
John Wall
Bradley Beal
Ian Mahinmi
Marcin Gortat
Bojan Bogdanovic
Otto Porter Jr
Tomas Satoransky
Markieff Morris
Marcus Thornton
Kelly Oubre Jr
Marcus Morris
−0.6 −0.4 −0.2 0.0 0.2 0.4 0.6
θ

```python
grid = sns.FacetGrid(time_df, col='team', col_wrap=2,
                     sharey=False,
                     size=4, aspect=1.5)
grid.map(plot_params_helper,
         'b_mean', 'b_low', 'b_high', 'name',
         league_mean=0.,
         league_mean_name="League average");

grid.set_axis_labels(r"$b$", "Player");

grid.fig.tight_layout();
grid.add_legend();
```



Jonathan Gibson
Quincy Acy
Dwight Powell
Seth Curry
Emmanuel Mudiay
Gary Harris
Wilson Chandler
team = DET
Aron Baynes
Reggie Jackson
Jon Leuer
Kentavious Caldwell-Pope
Reggie Bullock
Andre Drummond
Ish Smith
Stanley Johnson
Tobias Harris
Marcus Morris
Player
team = GSW
Klay Thompson
Stephen Curry
Draymond Green
Shaun Livingston
Andre Iguodala
Kevin Durant
Zaza Pachulia
James Michael McAdoo
Matt Barnes
team = HOU
Nene
Ryan Anderson
James Harden
Eric Gordon
Trevor Ariza
Corey Brewer
Clint Capela
KJ McDaniels
Patrick Beverley
Sam Dekker
Montrezl Harrell
Player
team = IND
Paul George
Joe Young
Glenn Robinson III
Lavoy Allen
Thaddeus Young
Kevin Seraphin
Jeff Teague
Rodney Stuckey
Myles Turner
George Hill
Aaron Brooks
CJ Miles
Monta Ellis
team = LAC
DeAndre Jordan
Blake Griffin
Jamal Crawford
Austin Rivers
Luc Mbah a Moute
Wesley Johnson
Paul Pierce
JJ Redick
Brandon Bass
Raymond Felton
Marreese Speights
Chris Paul
Player
team = LAL
Tarik Black
Thomas Robinson
D'Angelo Russell
Brandon Ingram
David Nwaba
Tyler Ennis
Timofey Mozgov
Lou Williams
Jordan Clarkson
Julius Randle
Luol Deng
Nick Young
Ivica Zubac
Larry Nance Jr
team = MEM
Marc Gasol
Tony Allen
Terrence Jones
Zach Randolph
Toney Douglas
JaMychal Green
Troy Williams
Vince Carter
Mike Conley
Jarell Martin
Andrew Harrison
Troy Daniels
James Ennis III
Player
team = MIA
Okaro White
Hassan Whiteside
Goran Dragic
Dion Waiters
Willie Reed
Rodney McGruder
Udonis Haslem
James Johnson
Luke Babbitt
Josh McRoberts
Justise Winslow
Wayne Ellington
Josh Richardson
Tyler Johnson
League average
Mean with
90% interval
team = MIL
Greg Monroe
Giannis Antetokounmpo
Thon Maker
Matthew Dellavedova
Khris Middleton
Tony Snell
John Henson
Miles Plumlee
Jason Terry
Mirza Teletovic
Player
team = MIN
Tyus Jones
Kris Dunn
Brandon Rush
Gorgui Dieng
Karl-Anthony Towns
Andre Iguodala
Andrew Wiggins
Nemanja Bjelica


Michael Beasley
Malcolm Brogdon
Ersan Ilyasova
Jabari Parker
Shabazz Muhammad
Zach LaVine
Ricky Rubio
team = NOP
Anthony Davis
Terrence Jones
Omer Asik
DeMarcus Cousins
Donatas Motiejunas
Alexis Ajinca
Dante Cunningham
Maurice Harkless
Tim Frazier
Jordan Crawford
Solomon Hill
Lance Stephenson
Anthony Brown
Tyreke Evans
Jrue Holiday
E'Twaun Moore
Buddy Hield
Langston Galloway
Player
team = NYK
Carmelo Anthony
Derrick Rose
Ron Baker
Joakim Noah
Mindaugas Kuzminskas
Kyle O'Quinn
Lance Thomas
Willy Hernangomez
Kristaps Porzingis
Justin Holiday
Brandon Jennings
Sasha Vujacic
Courtney Lee
team = OKC
Steven Adams
Taj Gibson
Russell Westbrook
Joffrey Lauvergne
Grant, Anthony
Semaj Christon
Domantas Sabonis
Enes Kanter
Anthony Morrow
Andre Roberson
Doug McDermott
Victor Oladipo
Kevin Durant
Alex Abrines
Jerami Grant
Player
team = ORL
Nikola Vucevic
Evan Fournier
Terrence Ross
Goran Dragic
Bismack Biyombo
Forcier, Chad
CJ Watson
Elfrid Payton
DJ Augustin
Serge Ibaka
Victor Oladipo
Jodie Meeks
Aaron Gordon
Jeff Green
team = PHI
Jahlil Okafor
Dario Saric
Nerlens Noel
Sergio Rodriguez
Jerryd Bayless
Timothe Luwawu-Cabarrot
TJ McConnell
Richaun Holmes
Justin Anderson
Joel Embiid
Hollis Thompson
Nik Stauskas
Gerald Henderson
Robert Covington
Ersan Ilyasova
Player
team = PHO
Tyson Chandler
Marquese Chriss
Brandon Knight
Alex Len
Dragan Bender
Leandro Barbosa
Eric Bledsoe
PJ Tucker
Alan Williams
Tyler Ulis
Devin Booker
TJ Warren
Jared Dudley
team = POR
Al-Farouq Aminu
Evan Turner
Jusuf Nurkic
Meyers Leonard
Mason Plumlee
Shabazz Napier
Maurice Harkless
Ed Davis
Pat Connaughton
Allen Crabbe
CJ McCollum
Damian Lillard
Player
team = SAC
Willie Cauley-Stein
Anthony Tolliver
DeMarcus Cousins
Ben McLemore
Kosta Koufos
Skal Labissiere
Arron Afflalo
Garrett Temple
Tyreke Evans
Omri Casspi
Ty Lawson
Justise Winslow
Buddy Hield
Matt Barnes
Rudy Gay
Darren Collison
Langston Galloway
team = SAS
LaMarcus Aldridge
Kawhi Leonard
Paul Millsap
Danny Green
David Lee
Dewayne Dedmon
Kyle Anderson
Dejounte Murray
Davis Bertans
Player
team = TOR
Patrick Patterson
DeMarre Carroll
Terrence Ross
Cory Joseph
Jonas Valanciunas
Fred VanVleet
Delon Wright
DeMar DeRozan
Norman Powell



Salary model

Our final model uses salary as a proxy for “star power” to explore its influence on foul calls. We also (somewhat naively) impute missing salaries to the (log) league average.

```
std_log_salary = (salary_df.ix[np.arange(n_players)]
                           .std_log_salary
                           .fillna(0)
                           .values)
```

With s_i denoting the i-the player’s standardized log salary, our model becomes

$$
\begin{aligned}
\theta_i &= \theta_{0,i} + \delta_\theta \cdot s_i \\
b_j &= b_{0,j} + \delta_b \cdot s_j \\
\eta_k &= \lambda_{t(k)} + \theta_{i(k)} - b_{j(k)}.
\end{aligned}
$$

In this model, each player’s θ_i and b_j parameters are linear functions of their standardized log salary, with (hierarchical) varying intercepts. The varying intercepts $\theta_{0,i}$ and $b_{0,j}$ are endowed with the same hierarchical normal priors as θ_i and b_j had in the previous model. We place normal priors, $\delta_\theta \sim N(0, 100)$ and $\delta_b \sim N(0, 100)$, on the salary coefficients.

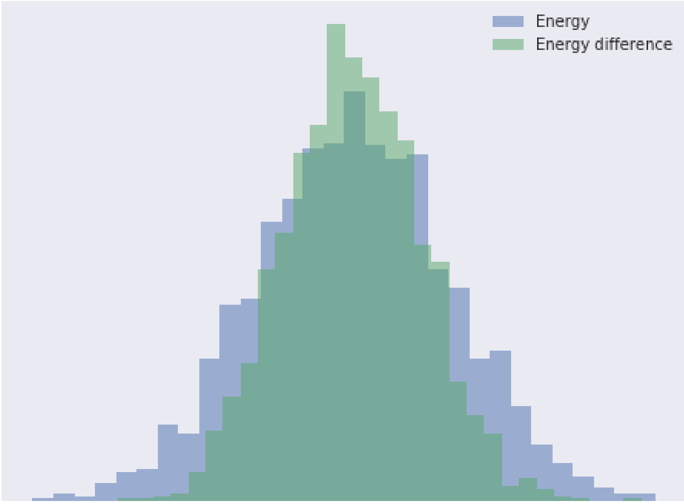
```
with pm.Model() as salary_model:
    τ_λ = pm.Exponential('τ_λ', 1e-4)
    λ = pm.GaussianRandomWalk('λ', tau=τ_λ,
                              init=pm.Normal.dist(0., 10.),
                              shape=n_sec)

    Δ_θ0 = pm.Normal('Δ_θ0', 0., 1., shape=n_players)
    σ_θ0 = pm.HalfCauchy('σ_θ0', 2.5)
    θ0 = pm.Deterministic('θ0', Δ_θ0 * σ_θ0)

    δ_θ = pm.Normal('δ_θ', 0., 10.)

    θ = pm.Deterministic('θ', θ0 + δ_θ * std_log_salary)

    Δ_b0 = pm.Normal('Δ_b0', 0., 1., shape=n_players)
    σ_b0 = pm.HalfCauchy('σ_b0', 2.5)
    b0 = pm.Deterministic('b0', Δ_b0 * σ_b0)

    δ_b = pm.Normal('δ_b', 0., 10.)

    b = pm.Deterministic('b', b0 + δ_b * std_log_salary)

    η = λ[sec] + θ[disadvantaged_id] - b[committing_id]
    p = pm.Deterministic('p', pm.math.sigmoid(η))

    y = pm.Bernoulli('y_obs', p, observed=foul_called)
```

```
salary_trace = sample(salary_model, N_TUNE, N_SAMPLES, SEED)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using advi...
Average ELBO = -1.0244e+05:  15%|█          | 30998/200000 [00:32<02:57, 952.52it/s]Median ELBO
 converged.
Finished [100%]: Average ELBO = -2,995.3

100%|██████████| 4000/4000 [03:45<00:00, 17.76it/s]
```

The energy plot for this model looks a bit worse than that for the time remaining model.

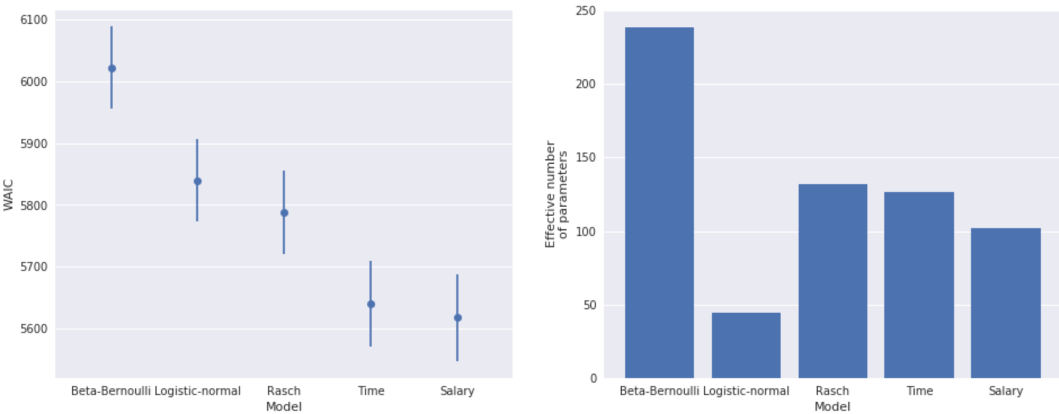
```
energy_plot(salary_trace)
```



The salary model also appears to be a slight improvement over the time remaining model in terms of WAIC.

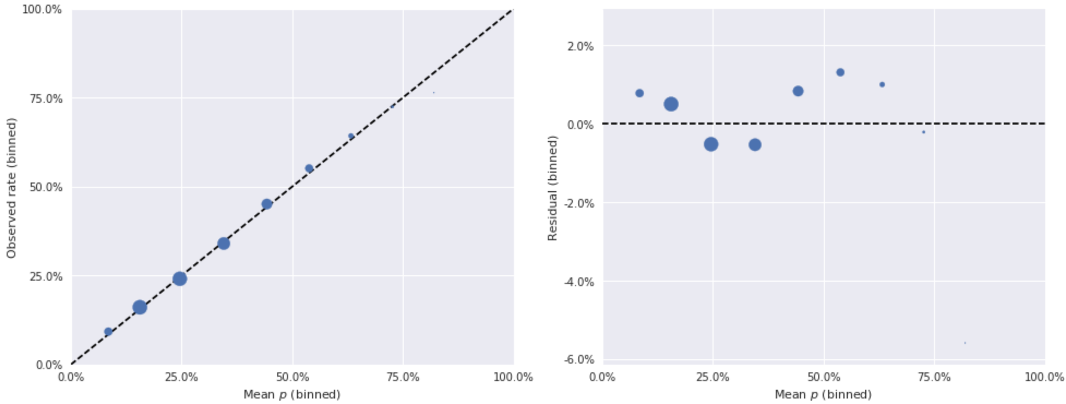
```
waic_df = waic_df.append(get_waic_df(salary_model, salary_trace, "Salary"))
```

```
waic_plot(waic_df)
```



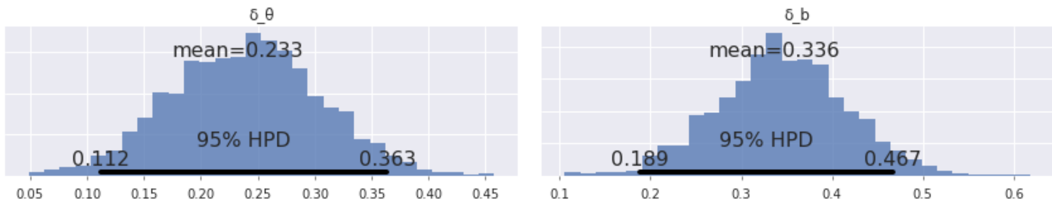
The binned residuals continue to look good for this model.

```
bin_obs, bin_p, bin_counts = binned_residuals(foul_called, salary_trace['p'])

binned_residual_plot(bin_obs, bin_p, bin_counts)
```



Based on the posterior distributions of δ_θ and δ_b, we expect to see a fairly strong relationship between a player's (standardized log) salary and their latent skill parameters.

```
pm.plot_posterior(salary_trace, ['δ_θ', 'δ_b'],
                  lw=0., alpha=0.75);
```



The following plots confirm this relationship.

```
salary_df_ = to_param_df(team_player_map, salary_trace, ['θ', 'θ0', 'b', 'b0'])
```

```
fig, (θ_ax, b_ax) = plt.subplots(ncols=2, sharex=True, figsize=(16, 6))

salary = (salary_df.ix[np.arange(n_players)]
                   .salary
                   .fillna(salary_df.salary.mean())
                   .values)

θ_ax.scatter(salary[salary_df_.player_id], salary_df_.θ_mean,
             alpha=0.75);

θ_ax.xaxis.set_major_formatter(million_dollars_formatter);
θ_ax.set_xlabel("Salary");

θ_ax.set_ylabel(r"$\theta$");

b_ax.scatter(salary[salary_df_.player_id], salary_df_.b_mean,
             alpha=0.75);

b_ax.xaxis.set_major_formatter(million_dollars_formatter);
b_ax.set_xlabel("Salary");

b_ax.set_ylabel(r"$b$");
```