1 Mathematical Formulation of Fan Vote Inference

The core of our problem is to infer the unobserved fan voting preferences based on the observed judges' scores and the weekly elimination results. We model each week within each season as an independent Bayesian inference problem.

1.1 Notations and Problem Definition

Let s denote the season index and w denote the week index. We define the set of contestants participating in season s, week w as $\mathcal{C}_{s,w}$, with the total number of contestants $N_{s,w} = |\mathcal{C}_{s,w}|$.

For each contestant $i \in \mathcal{C}_{s,w}$, we define the following variables:

- **Observed Variable (Judges):** Let $J_{s,w,i} \in \mathbb{R}^+$ be the total score given by the panel of judges to contestant i. When context is clear, we abbreviate this as J_i.
- **Latent Variable (Fans):** Let $\theta_{s,w,i} \in [0, 1]$ represent the *true fan vote share* (proportion of total fan votes) received by contestant i. This is the unknown parameter we aim to estimate.
- **Elimination Outcome:** Let $E_{s,w} \in \mathcal{C}_{s,w}$ denote the contestant who was eliminated at the end of week w. This serves as the primary constraint for our inference.

Since the exact number of fan votes is unknown and varies significantly across weeks and seasons, we focus on the *share* of votes rather than absolute counts. The vector of fan vote shares $\boldsymbol{\theta}_{s,w} = (\theta_{s,w,1}, \ldots, \theta_{s,w,N_{s,w}})$ must satisfy the **simplex constraint**:

$$\sum_{i=1}^{N_{s,w}} \theta_{s,w,i} = 1, \quad \text{and} \quad \theta_{s,w,i} \geq 0 \quad \forall i. \tag{1}$$

This constraint ensures that fan vote shares form a valid probability distribution over contestants, making the Dirichlet distribution a natural modeling choice.

1.2 Prior Distribution

We adopt a symmetric Dirichlet prior for the fan vote shares:

$$\boldsymbol{\theta}_{s,w} \sim \text{Dirichlet}(\boldsymbol{\alpha}), \quad \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N) = (1, 1, \ldots, 1). \tag{2}$$

The uniform Dirichlet prior ($\alpha_i = 1$ for all i) represents maximum ignorance about the relative popularity of contestants before observing elimination outcomes. The probability density function is:

$$p(\boldsymbol{\theta}) = \frac{\Gamma(\sum_{i=1}^{N} \alpha_i)}{\prod_{i=1}^{N} \Gamma(\alpha_i)} \prod_{i=1}^{N} \theta_i^{\alpha_i - 1} = (N-1)! \cdot \mathbf{1}_{\Delta^{N-1}}(\boldsymbol{\theta}), \tag{3}$$

where Δ^{N-1} denotes the $(N-1)$-dimensional probability simplex.

1.3 Likelihood Functions and Elimination Constraints

The observed data for each week is the identity of the eliminated contestant $E_{s,w}$. The likelihood function $P(E_{s,w}|\boldsymbol{\theta}_{s,w})$ is determined by the specific elimination rules $R(s)$ active during season s. Since the exact vote counts are unavailable, we formulate the likelihood as a constraint satisfaction problem.

Conceptually, the likelihood takes the form:

$$\mathcal{L}(\boldsymbol{\theta}_{s,w}) \propto \begin{cases} 1 & \text{if } \boldsymbol{\theta}_{s,w} \text{ creates a ranking consistent with } E_{s,w} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

However, hard indicator constraints prevent gradient-based MCMC sampling. We therefore relax these constraints using continuous potential functions, as detailed below for each rule era.

1.3.1 Era 1: Rank-Based Points System (Seasons 1–2)

In the earliest seasons, contestants accumulate points based on their performance rankings rather than raw scores.

Judge Points Calculation. Let J_i denote the raw judge score for contestant i. We first compute the rank $r^J(i)$ where the contestant with the highest score receives rank 1. The judge points are then assigned inversely:

$$P_i^J = N - r^J(i) + 1, \tag{5}$$

so that the highest-scoring contestant receives N points and the lowest-scoring receives 1 point.

We normalize these points to obtain a proportion:

$$P_i^{J,\text{norm}} = \frac{P_i^J}{\sum_{k=1}^{N} P_k^J}. \tag{6}$$

Combined Score. The total score for contestant i combines the normalized judge points with the fan vote share:

$$S_i = P_i^{J,\text{norm}} + \theta_{s,w,i}. \tag{7}$$

Note that both terms are normalized proportions, so $S_i \in [0, 2]$ and $\sum_i S_i = 2$.

Elimination Constraint. The eliminated contestant E must have the strictly lowest total score among all contestants:

$$\mathcal{C}_{\text{Era1}}: \quad S_E < S_k, \quad \forall k \in \mathcal{C}_{s,w} \setminus \{E\}. \tag{8}$$

Soft Constraint Relaxation. To enable MCMC sampling, we replace the hard inequality with a log-sigmoid potential:

$$\log \mathcal{L}_{\text{Era1}}(\boldsymbol{\theta}) = \sum_{k \neq E} \log \sigma \left(\lambda \cdot (S_k - S_E) \right), \tag{9}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is the logistic sigmoid function and $\lambda = 100$ is a scaling factor that controls the sharpness of the constraint. As $\lambda \to \infty$, this approaches the hard indicator function.

Tie-Breaker Mechanism. When two contestants have nearly identical total scores, the show's actual rules use fan votes as the tie-breaker (lower fan share is eliminated). We incorporate this with an additional potential:

$$\phi_{\text{tie}}(i, E) = \mathbf{1}_{|S_i - S_E| < \epsilon} \cdot \log \sigma \left(\lambda \cdot (\theta_i - \theta_E) \right), \tag{10}$$

where the indicator $\mathbf{1}_{|S_i - S_E| < \epsilon}$ is approximated by a soft sigmoid:

$$\mathbf{1}_{|S_i - S_E| < \epsilon} \approx \sigma(-1000 \cdot |S_i - S_E| + 5). \tag{11}$$

The complete log-likelihood for Era 1 is:

$$\log \mathcal{L}_{\text{Era1}}(\boldsymbol{\theta}) = \sum_{k \neq E} \left[\log \sigma(\lambda(S_k - S_E)) + \phi_{\text{tie}}(k, E) \right]. \tag{12}$$

1.3.2 Era 2: Percentage-Based System (Seasons 3–27)

From Season 3 onwards, the show adopted a simpler percentage-based system where raw judge percentages are directly combined with fan vote shares.

Normalized Judge Score. Let J_i be the raw judge score. The normalized judge percentage is:

$$J_i^{\text{norm}} = \frac{J_i}{\sum_{k=1}^{N} J_k}. \tag{13}$$

Combined Score. The total score is the sum of judge and fan percentages:

$$S_i = J_i^{\text{norm}} + \theta_{s,w,i}. \tag{14}$$

Elimination Constraint. The eliminated contestant must have the lowest combined percentage:

$$\mathcal{C}_{\text{Era2}}: \quad S_E < S_k, \quad \forall k \in \mathcal{C}_{s,w} \setminus \{E\}. \tag{15}$$

Soft Constraint Relaxation. The log-likelihood contribution is:

$$\log \mathcal{L}_{\text{Era2}}(\boldsymbol{\theta}) = \sum_{k \neq E} \log \sigma \left(\lambda \cdot (S_k - S_E) \right), \tag{16}$$

with $\lambda = 100$.

1.3.3 Era 3: Bottom-Two Rule (Seasons 28+)

Beginning with Season 28, the elimination process changed significantly. The judges now select which contestant to eliminate from the "Bottom Two" couples—those with the two lowest combined scores. Crucially, within the Bottom Two, the final elimination is determined by *live judge voting*, not by score. This means:

1. The eliminated contestant must be *in* the Bottom Two ($|\mathcal{K}_{\text{lower}}| \leq 1$).
2. Both $|\mathcal{K}_{\text{lower}}| = 0$ (lowest score) and $|\mathcal{K}_{\text{lower}}| = 1$ (second-lowest) are equally valid outcomes.

Informative Prior. Unlike Eras 1–2, we employ an *informative* Dirichlet prior for Era 3. The rationale is that fan voting patterns tend to correlate with judge scores—contestants who perform well typically receive more fan support. We parameterize:

$$\alpha_i = 0.5 + \kappa \cdot \frac{r^J(i) - 1}{N - 1}, \tag{17}$$

where $r^J(i)$ is the rank of contestant i based on judge scores (1 = lowest, N = highest), and $\kappa = 5.0$ is the prior concentration strength. This yields $\alpha_i \in [0.5, 5.5]$, with higher-scoring contestants receiving higher prior expected fan shares.

Combined Score. The total score follows the same structure as Era 1:

$$S_i = P_i^{J,\text{norm}} + \theta_{s,w,i}. \tag{18}$$

Bottom-Two Constraint. Define the set of survivors with scores strictly lower than the eliminated contestant:

$$\mathcal{K}_{\text{lower}} = \{k \in \mathcal{C}_{s,w} \setminus \{E\} \mid S_k < S_E\}. \tag{19}$$

The constraint requires that at most one survivor has a lower score than E:

$$\mathcal{C}_{\text{B2}} : \quad |\mathcal{K}_{\text{lower}}| \leq 1. \tag{20}$$

Soft Cardinality Approximation. We approximate the cardinality using a sum of sigmoid functions with a *softer* scale parameter:

$$\widehat{|\mathcal{K}_{\text{lower}}|} = \sum_{k \neq E} \sigma(\lambda_{\text{soft}} \cdot (S_E - S_k)), \tag{21}$$

where $\lambda_{\text{soft}} = 20$ (reduced from $\lambda = 100$ used in Eras 1–2). The softer boundary allows the posterior to explore configurations where the eliminated contestant is either the lowest or second-lowest scorer.

Quadratic Penalty with Tolerance. Violations of the Bottom-Two constraint are penalized quadratically, with an explicit tolerance threshold:

$$\log \mathcal{L}_{\text{B2}}(\boldsymbol{\theta}) = -\gamma \cdot \Big(\max(0, |\widehat{\mathcal{K}_{\text{lower}}}| - \tau)\Big)^2, \tag{22}$$

where $\gamma = 20$ is the penalty strength and $\tau = 1.3$ is the tolerance threshold. This formulation:

- Allows $|\mathcal{K}_{\text{lower}}| \leq 1.3$ without penalty (accommodating sigmoid approximation noise).
- Applies gentle quadratic penalty when clearly outside Bottom Two.
- Does *not* favor lowest score over second-lowest, reflecting the judge vote randomness.

Parameter	Symbol	Value
Prior concentration	κ	5.0
Soft constraint scale	λ_{soft}	20.0
Penalty threshold	τ	1.3
Penalty strength	γ	20.0

Table 1: Era 3 (Bottom-Two Rule) model parameters. These settings allow balanced exploration of both $|\mathcal{K}_{\text{lower}}| = 0$ and $|\mathcal{K}_{\text{lower}}| = 1$ outcomes.

Era 3 Parameter Summary.

1.4 Posterior Distribution

Combining the Dirichlet prior with the rule-specific likelihood, the posterior distribution is:

$$p(\boldsymbol{\theta}_{s,w}|E_{s,w}) \propto p(\boldsymbol{\theta}_{s,w}) \cdot \mathcal{L}_{R(s)}(\boldsymbol{\theta}_{s,w}), \tag{23}$$

where $R(s) \in \{\text{Era1}, \text{Era2}, \text{B2}\}$ denotes the rule type for season s.

In log-space:

$$\log p(\boldsymbol{\theta}|E) = \log p(\boldsymbol{\theta}) + \log \mathcal{L}_{R(s)}(\boldsymbol{\theta}), \tag{24}$$

where:

- For Eras 1–2: $\log p(\boldsymbol{\theta}) = \text{const}$ (uniform Dirichlet prior).
- For Era 3: $\log p(\boldsymbol{\theta}) = \sum_{i=1}^{N} (\alpha_i - 1) \log \theta_i + \text{const}$ (informative prior correlated with judge scores).

1.5 MCMC Sampling Strategy

We employ Markov Chain Monte Carlo (MCMC) methods to sample from the posterior distribution.

1.5.1 Sampler Selection

The choice of sampler depends on the differentiability of the log-posterior:

- **Era 2 (Percentage-based):** The log-likelihood is smooth and differentiable. We use the No-U-Turn Sampler (NUTS), an adaptive Hamiltonian Monte Carlo variant that automatically tunes step size and trajectory length.
- **Era 1 and Era 3:** The rank-based computations and discrete-like tie-breaker mechanisms introduce numerical challenges. We use the Slice Sampler, which does not require gradient information and handles multimodal posteriors more robustly.

1.5.2 Sampling Parameters

Parameter	Standard	Enhanced (S1–S2)
Number of draws	2000	5000
Tuning iterations	1000	2000
Number of chains	2	4
Sampler	NUTS / Slice	Slice

Table 2: MCMC sampling parameters. Enhanced settings are used for Seasons 1–2 to ensure convergence given the additional tie-breaker constraints.

1.5.3 Enhanced Sampling for Seasons 1–2

The Era 1 model (Seasons 1–2) presents unique computational challenges that necessitate an enhanced sampling strategy. We implement a specialized sampling procedure with significantly increased computational resources.

Challenges in Era 1 Inference. Three factors contribute to the difficulty of sampling from the Era 1 posterior:

1. **Tie-Breaker Discontinuity:** The tie-breaker potential ϕ_{tie} introduces near-discontinuous behavior in the log-posterior. When $|S_i - S_E|$ crosses the threshold $\epsilon \approx 0.005$, the potential changes rapidly, creating sharp ridges in the posterior landscape.
2. **Rank-Based Discretization:** Unlike the percentage-based system, the rank transformation $r^J(i)$ is inherently discrete. While we work with normalized points, the underlying rank structure creates non-smooth regions where small changes in fan shares can cause rank swaps.
3. **Constraint Interactions:** The combination of elimination constraints and tie-breaker constraints creates a complex feasible region. The posterior mass concentrates in narrow corridors of the simplex where both constraints are satisfied.

Slice Sampler. For Era 1 (and Era 3), we employ the Slice Sampler rather than gradient-based methods. The Slice Sampler operates by:

1. Given current state $\boldsymbol{\theta}^{(t)}$, sample auxiliary variable $u \sim \text{Uniform}(0, p(\boldsymbol{\theta}^{(t)}|E))$.
2. Define the "slice" $\mathcal{S} = \{\boldsymbol{\theta} : p(\boldsymbol{\theta}|E) > u\}$.
3. Sample $\boldsymbol{\theta}^{(t+1)}$ uniformly from $\mathcal{S}$ using stepping-out and shrinkage procedures.

The Slice Sampler is gradient-free and handles multimodal or irregular posteriors more robustly than Hamiltonian Monte Carlo, at the cost of higher autocorrelation.

Enhanced Parameter Settings. To compensate for the increased autocorrelation and ensure adequate exploration, we use enhanced sampling parameters for Seasons 1–2:

Parameter	Standard (Era 2)	Enhanced (Era 1)	Ratio
Number of draws N_{draw}	2000	5000	2.5×
Tuning iterations N_{tune}	1000	2000	2×
Number of chains M	2	4	2×
Total samples	4000	20000	5×

Table 3: Comparison of standard and enhanced MCMC parameters. The enhanced settings provide 5× more total samples to ensure convergence despite higher autocorrelation.

Effective Sample Size Decomposition. We monitor two variants of effective sample size:

- **ESS-bulk** (ESS_{bulk}): Measures the effective number of independent samples for estimating the posterior mean and central credible intervals. Computed using rank-normalized draws to improve robustness.
- **ESS-tail** (ESS_{tail}): Measures the effective sample size for estimating tail quantiles (5% and 95%). This is typically lower than ESS-bulk because tail regions are visited less frequently.

For reliable inference, we require:

$$\text{ESS}_{\text{bulk}} > 400, \quad \text{ESS}_{\text{tail}} > 400. \tag{25}$$

Convergence Criteria. A week is considered "converged" if all of the following hold:

$$\max_i \hat{R}(\theta_i) < 1.05, \tag{26}$$

$$\min_i \text{ESS}_{\text{bulk}}(\theta_i) > 400, \tag{27}$$

$$\min_i \text{ESS}_{\text{tail}}(\theta_i) > 400. \tag{28}$$

With the enhanced sampling strategy, we achieve 100% convergence across all Season 1–2 weeks.

1.5.4 Convergence Diagnostics

We assess convergence using two standard diagnostics:

$\hat{R}$ (R-hat) Statistic. The potential scale reduction factor compares between-chain and within-chain variance:

$$\hat{R} = \sqrt{\frac{\hat{V}}{W}}, \tag{29}$$

where $\hat{V}$ is the estimated marginal posterior variance and W is the within-chain variance. We require $\hat{R} < 1.05$ for all parameters.

Effective Sample Size (ESS). ESS accounts for autocorrelation in the MCMC chains:

$$\text{ESS} = \frac{MN}{1 + 2\sum_{t=1}^{\infty} \rho_t}, \tag{30}$$

where M is the number of chains, N is the number of draws per chain, and ρ_t is the autocorrelation at lag t. We require ESS > 400 for reliable inference.

1.6 Point Estimates and Uncertainty Quantification

1.6.1 Posterior Mean

The point estimate for each contestant's fan vote share is the posterior mean:

$$\hat{\theta}_i = \mathbb{E}[\theta_i | E_{s,w}] \approx \frac{1}{T}\sum_{t=1}^{T} \theta_i^{(t)}, \tag{31}$$

where $\{\theta_i^{(t)}\}_{t=1}^{T}$ are the MCMC samples after discarding burn-in.

1.6.2 Highest Density Interval (HDI)

We report 95% HDI as the credible interval:

$$\text{HDI}_{95\%}(\theta_i) = [L_i, U_i], \tag{32}$$

where $[L_i, U_i]$ is the shortest interval containing 95% of the posterior probability mass. The HDI width $U_i - L_i$ quantifies estimation uncertainty.

1.7 Self-Consistency Verification

As a diagnostic check, we verify that posterior mean estimates satisfy the elimination constraints. Since the likelihood function explicitly enforces these constraints (via soft potentials with $\lambda = 100$), self-consistency is a necessary condition for convergence rather than an independent validation metric.

Verification Criterion. For each week, we check whether the posterior mean $\hat{\boldsymbol{\theta}}$ produces a combined score ranking consistent with the observed elimination:

$$\text{Consistent}_{s,w} = \mathbf{1}\left[\hat{S}_E \leq \hat{S}_{(2)}\right], \tag{33}$$

where $\hat{S}_{(2)}$ denotes the second-lowest combined score among all contestants. This unified criterion covers both the "lowest score" rule (Eras 1–2) and the "Bottom-Two" rule (Era 3).

A week failing this check indicates MCMC non-convergence or numerical issues, requiring re-sampling with enhanced parameters.

1.8 Algorithm Summary

Algorithm 1 Bayesian Fan Vote Share Inference (General)

Require: Season s, week w, contestants $\mathcal{C}_{s,w}$, judge scores $\{J_i\}$, eliminated contestant E
Ensure: Posterior samples $\{\boldsymbol{\theta}^{(t)}\}_{t=1}^{T}$, diagnostics

```
  Determine rule type: R(s) ← { Era1  s ≤ 2 ; Era2  3 ≤ s ≤ 27 ; B2  s ≥ 28 }
  Compute normalized scores:
  if R(s) = Era2 then
      P_i^norm ← J_i / Σ_k J_k                                   ▷ Direct percentage
  else
      Compute ranks r^J(i) from {J_i}
      P_i^J ← N − r^J(i) + 1                                      ▷ Rank to points
      P_i^norm ← P_i^J / Σ_k P_k^J
  end if
  Initialize: θ ~ Dirichlet(1_N)
  Define log-likelihood log L_{R(s)}(θ) per Section 1.3
  Select sampler and parameters:
  if R(s) = Era1 then
      sampler ← Slice, (N_draw, N_tune, M) ← (5000, 2000, 4)     ▷ Enhanced
  else if R(s) = Era2 then
      sampler ← NUTS, (N_draw, N_tune, M) ← (2000, 1000, 2)
  else
      sampler ← Slice, (N_draw, N_tune, M) ← (2000, 1000, 2)
  end if
  Run MCMC with M parallel chains, each with N_tune tuning + N_draw sampling iterations
  Compute diagnostics: R̂, ESS_bulk, ESS_tail
  return {θ^(t)}_{t=1}^T, diagnostics
```

Algorithm 2 Era 1 Enhanced Sampling (Seasons 1–2 Specialized)

Require: Week data with N contestants, eliminated contestant E
Ensure: Posterior samples with verified convergence
  Step 1: Compute Judge Points
  for each contestant i **do**
    $r^J(i) \leftarrow$ rank of J_i (highest score = rank 1)
    $P_i^J \leftarrow N - r^J(i) + 1$
  end for
  $P_i^{J,\text{norm}} \leftarrow P_i^J / \sum_k P_k^J$ for all i
  Step 2: Define Potentials
  Initialize log-likelihood: $\ell(\boldsymbol{\theta}) \leftarrow 0$
  for each survivor $k \neq E$ **do**
    $S_k \leftarrow P_k^{J,\text{norm}} + \theta_k, \quad S_E \leftarrow P_E^{J,\text{norm}} + \theta_E$
    $\ell \leftarrow \ell + \log \sigma(\lambda(S_k - S_E))$ ▷ Elimination constraint
    $\delta \leftarrow |S_k - S_E|$
    $w_{\text{tie}} \leftarrow \sigma(-1000 \cdot \delta + 5)$ ▷ Tie indicator
    $\ell \leftarrow \ell + w_{\text{tie}} \cdot \log \sigma(\lambda(\theta_k - \theta_E))$ ▷ Tie-breaker
  end for
  Step 3: Enhanced MCMC Sampling
  Initialize $M = 4$ chains from Dirichlet($\mathbf{1}_N$)
  for each chain $m = 1, \ldots, 4$ **do**
    Run $N_{\text{tune}} = 2000$ tuning iterations (Slice sampler)
    Run $N_{\text{draw}} = 5000$ sampling iterations
    Store samples $\{\boldsymbol{\theta}^{(m,t)}\}_{t=1}^{5000}$
  end for
  Step 4: Convergence Verification
  Compute $\hat{R}_i$ for each θ_i across 4 chains
  Compute $\text{ESS}_{\text{bulk},i}$ and $\text{ESS}_{\text{tail},i}$
  converged $\leftarrow (\max_i \hat{R}_i < 1.05) \wedge (\min_i \text{ESS}_{\text{bulk},i} > 400)$
  return All $4 \times 5000 = 20000$ posterior samples, convergence status

1.9 Notation Summary

1.10 Implementation Notes

The model is implemented in Python using PyMC (version 5.x) for probabilistic programming and ArviZ for diagnostics. Key implementation details:

- **Dirichlet Parameterization:** We use PyMC's native `Dirichlet` distribution with concentration $\boldsymbol{\alpha} = \mathbf{1}_N$, which automatically handles the simplex constraint.
- **Potential Functions:** Soft constraints are implemented via `pm.Potential`, which adds arbitrary log-probability terms to the model.
- **Tensor Operations:** Pytensor (formerly Theano) provides automatic differentiation for NUTS and efficient array operations for the Slice sampler.

- **Random Seed:** All sampling uses a fixed random seed (42) for reproducibility.
- **Rank Computation:** The `scipy.stats.rankdata` function with `method='average'` handles tied judge scores by assigning the average rank.

Symbol	Description
Data and Indices	
s, w	Season and week indices
$N_{s,w}$	Number of contestants in week w of season s
$\mathcal{C}_{s,w}$	Set of contestants
$E_{s,w}$	Eliminated contestant (observed)
Judge Scores	
J_i	Raw judge score for contestant i
$r^J(i)$	Rank of contestant i based on judge scores (1 = highest)
P_i^J	Judge points derived from rank: $P_i^J = N - r^J(i) + 1$
$P_i^{J,\text{norm}}$	Normalized judge points: $P_i^J / \sum_k P_k^J$
J_i^{norm}	Normalized judge percentage: $J_i / \sum_k J_k$
Fan Vote Model	
$\theta_{s,w,i}$	Latent fan vote share for contestant i
$\boldsymbol{\theta}_{s,w}$	Fan vote share vector $(\theta_1, \ldots, \theta_N)$
$\boldsymbol{\alpha}$	Dirichlet prior concentration parameters
S_i	Combined score: $P_i^{J,\text{norm}} + \theta_i$ or $J_i^{\text{norm}} + \theta_i$
Constraint Parameters	
λ	Constraint sharpness parameter ($\lambda = 100$)
γ	Penalty strength for Bottom-Two rule ($\gamma = 50$)
$\sigma(\cdot)$	Logistic sigmoid function: $\sigma(x) = (1 + e^{-x})^{-1}$
ϕ_{tie}	Tie-breaker potential function
MCMC Parameters	
N_{draw}	Number of posterior samples per chain
N_{tune}	Number of tuning/burn-in iterations
M	Number of parallel MCMC chains
T	Total number of posterior samples: $T = M \times N_{\text{draw}}$
Convergence Diagnostics	
$\hat{R}$	Potential scale reduction factor (R-hat)
ESS_{bulk}	Effective sample size for bulk of distribution
ESS_{tail}	Effective sample size for distribution tails
Output	
$\hat{\theta}_i$	Posterior mean estimate of fan vote share
$\text{HDI}_{95\%}$	95% Highest Density Interval

Table 4: Summary of notation used in the model.