Learning about social and political polarization using "How many X's do you know" surveys

Andrew Gelman
Dept of Statistics and Dept of Political Science
Columbia University

4 April 2005

Overview

- Social and political polarization
- "How many X's do you know" surveys
- 3 models and Bayesian inference
- Our research plan

- collaborators:
  - Tian Zheng, Dept of Statistics, Columbia University
  - Matt Salganik, Dept of Sociology, Columbia University
  - Tom DiPrete, Dept of Sociology, Columbia University
  - Julien Teitler, School of Social Work, Columbia University
  - Jouni Kerman, Dept of Statistics, Columbia University
  - Peter Killworth and Chris McCarty shared their survey data

Overview

- Social and political polarization
- "How many X's do you know" surveys
- 3 models and Bayesian inference
- Our research plan

- collaborators:
  - Tian Zheng, Dept of Statistics, Columbia University
  - Matt Salganik, Dept of Sociology, Columbia University
  - Tom DiPrete, Dept of Sociology, Columbia University
  - Julien Teitler, School of Social Work, Columbia University
  - Jouni Kerman, Dept of Statistics, Columbia University
  - Peter Killworth and Chris McCarty shared their survey data

Overview

- Social and political polarization
- "How many X's do you know" surveys
- 3 models and Bayesian inference
- Our research plan

- collaborators:
  - Tian Zheng, Dept of Statistics, Columbia University
  - Matt Salganik, Dept of Sociology, Columbia University
  - Tom DiPrete, Dept of Sociology, Columbia University
  - Julien Teitler, School of Social Work, Columbia University
  - Jouni Kerman, Dept of Statistics, Columbia University
  - Peter Killworth and Chris McCarty shared their survey data

Overview

- Social and political polarization
- "How many X's do you know" surveys
- 3 models and Bayesian inference
- Our research plan

- collaborators:
  - Tian Zheng, Dept of Statistics, Columbia University
  - Matt Salganik, Dept of Sociology, Columbia University
  - Tom DiPrete, Dept of Sociology, Columbia University
  - Julien Teitler, School of Social Work, Columbia University
  - Jouni Kerman, Dept of Statistics, Columbia University
  - Peter Killworth and Chris McCarty shared their survey data

Overview

- Social and political polarization
- "How many X's do you know" surveys
- 3 models and Bayesian inference
- Our research plan

- collaborators:
  - Tian Zheng, Dept of Statistics, Columbia University
  - Matt Salganik, Dept of Sociology, Columbia University
  - Tom DiPrete, Dept of Sociology, Columbia University
  - Julien Teitler, School of Social Work, Columbia University
  - Jouni Kerman, Dept of Statistics, Columbia University
  - Peter Killworth and Chris McCarty shared their survey data

Overview

- Social and political polarization
- "How many X's do you know" surveys
- 3 models and Bayesian inference
- Our research plan

- collaborators:
  - Tian Zheng, Dept of Statistics, Columbia University
  - Matt Salganik, Dept of Sociology, Columbia University
  - Tom DiPrete, Dept of Sociology, Columbia University
  - Julien Teitler, School of Social Work, Columbia University
  - Jouni Kerman, Dept of Statistics, Columbia University
  - Peter Killworth and Chris McCarty shared their survey data

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:
  - Later marriage, fewer children
  - "Bowling alone" (Putnam)
  - Less involvement in community groups, labor unions, ...

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:
  - Later marriage, fewer children
  - "Bowling alone" (Putnam)
  - Less involvement in community groups, labor unions,

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:
  - Later marriage, fewer children
  - "Bowling alone" (Putnam)
  - Less involvement in community groups, labor unions, . . .

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:
  - Later marriage, fewer children
  - "Bowling alone" (Putnam)
  - Less involvement in community groups, labor unions, ...

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:
  - Later marriage, fewer children
  - "Bowling alone" (Putnam)
  - Less involvement in community groups, labor unions, . . .

Increasing social/economic heterogeneity in U.S. since 1950s?

- Social polarization:
  - More variety in domestic arrangements
  - Greater income inequality
  - We tend to know people of similar social class to ourselves
  - Counter-trend: more interracial marriages
- Decline in social capital:
  - Later marriage, fewer children
  - "Bowling alone" (Putnam)
  - Less involvement in community groups, labor unions, . . .

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:
  - Democrats know Democrats, Republicans know Republicans
  - Partisanship is correlated with income, religiosity
  - Diffusion of information and attitudes through social networks

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:
  - Democrats know Democrats, Republicans know Republicans
  - Partisanship is correlated with income, religiosity
  - Diffusion of information and attitudes through social networks

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:
  - Democrats know Democrats, Republicans know Republicans
  - Partisanship is correlated with income, religiosity
  - Diffusion of information and attitudes through social networks

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:
  - Democrats know Democrats, Republicans know Republicans
  - Partisanship is correlated with income, religiosity
  - Diffusion of information and attitudes through social networks

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:
  - Democrats know Democrats, Republicans know Republicans
  - Partisanship is correlated with income, religiosity
  - Diffusion of information and attitudes through social networks

Increasing political polarization in U.S. since 1970s?

- Polarization in political opinions:
  - More extreme liberals, more extreme conservatives, fewer moderates
  - "Stubborn American voter" (Joe Bafumi): politics affects economic views
- Connection to economic and social networks:
  - Democrats know Democrats, Republicans know Republicans
  - Partisanship is correlated with income, religiosity
  - Diffusion of information and attitudes through social networks

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, ...)
  - GSS, NES questions on values (White, Brooks, ...)
  - Community surveys (Putnam, ...)
  - General Social Survey: questions about your close contacts (DiMaggio, ...)
- Political polarization

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, ...)
  - GSS, NES questions on values (White, Brooks, ...)
  - Community surveys (Putnam, ...)
  - General Social Survey: questions about your close contacts (DiMaggio, ...)
- Political polarization
  - Congressional votes (McCarty, Poole, Rosenthal, ...)
  - NES and commercial polls (Page and Shapiro, Baldassarri, ...)

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization
  - Congressional votes (McCarty, Poole, Rosenthal, . . .)
  - NES and commercial polls (Page and Shapiro, Bafumi, . . .)

Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization
  - Congressional votes (McCarty, Poole, Rosenthal, . . .)
  - NES and commercial polls (Page and Shapiro, Bafumi, . . .)

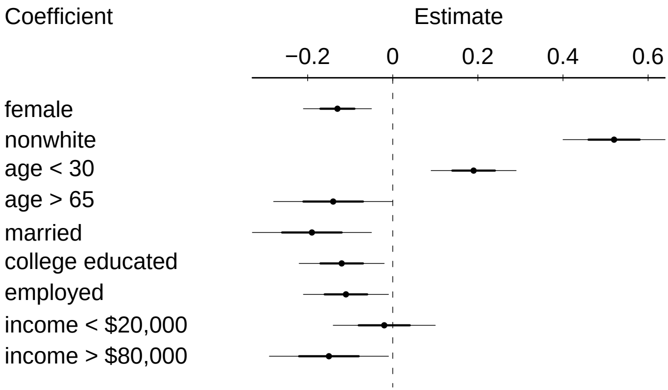
Past work studying polarization using surveys

- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization
  - Congressional votes (McCarty, Poole, Rosenthal, . . .)
  - NES and commercial polls (Page and Shapiro, Bafumi, . . .)

Past work studying polarization using surveys

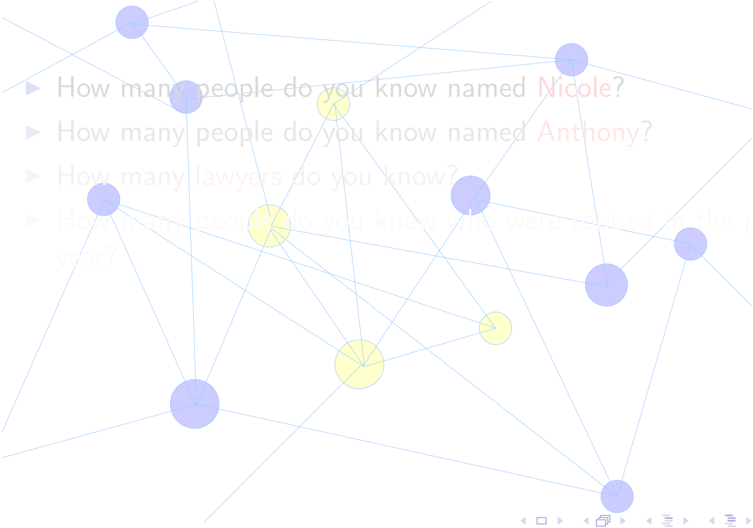
- Lots and lots has been done; this is an incomplete review
- Social polarization, social capital:
  - Census data on family characteristics (Cherlin, Mayer, Held, . . .)
  - GSS, NES questions on values (White, Brooks, . . .)
  - Community surveys (Putnam, . . .)
  - General Social Survey: questions about your close contacts (DiMaggio, . . .)
- Political polarization
  - Congressional votes (McCarty, Poole, Rosenthal, . . .)
  - NES and commercial polls (Page and Shapiro, Bafumi, . . .)

Example analysis: regression of residuals for "How many prisoners do you know?"



How many people do you know? Demonstration

- How many people do you know named Nicole?
- How many people do you know named Anthony?
- How many lawyers do you know?
- How many people do you know who were robbed in the past year?



How many people do you know? Demonstration

- How many people do you know named Nicole?
- How many people do you know named Anthony?
- How many lawyers do you know?
- How many people do you know who were robbed in the past year?

How many people do you know? Demonstration

- How many people do you know named Nicole?
- How many people do you know named Anthony?
- How many lawyers do you know?
- How many people do you know who were robbed in the past year?

How many people do you know? Demonstration

- How many people do you know named Nicole?
- How many people do you know named Anthony?
- How many lawyers do you know?
- How many people do you know who were robbed in the past year?

How many people do you know? Demonstration

- How many people do you know named Nicole?
- How many people do you know named Anthony?
- How many lawyers do you know?
- How many people do you know who were robbed in the past year?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Scale-up method: demonstration

- On average, you knew 0.6 Nicoles
- 0.13% of Americans are named Nicole
- Assume 0.13% of your acquaintances are Nicoles
- Estimate: on average, you know $0.6/0.0013 = 450$ people

- On average, you know 0.8 Anthonys
- 0.31% of Americans are named Anthony
- Estimate: on average, you know $1.6/0.0031 = 260$ people

- Why do these differ?

Estimating group sizes: demonstration

- On average, you know 2.6 lawyers
- Assume average network size is 450 people
- Estimate: lawyers represent $2.6/450 = 0.58\%$ of the network
- Estimate: $0.0058 \cdot 290\,\text{million} = 1.7$ million lawyers in the U.S.

- On average, you know 0.25 people who were robbed last year
- Estimate: $\frac{0.25}{450} \cdot 290\,\text{million} = 160{,}000$ people robbed

Estimating group sizes: demonstration

- On average, you know 2.6 lawyers
- Assume average network size is 450 people
- Estimate: lawyers represent $2.6/450 = 0.58\%$ of the network
- Estimate: $0.0058 \cdot 290\,\text{million} = 1.7$ million lawyers in the U.S.

- On average, you know 0.25 people who were robbed last year
- Estimate: $\frac{0.25}{450} \cdot 290\,\text{million} = 160{,}000$ people robbed

Estimating group sizes: demonstration

- On average, you know 2.6 lawyers
- Assume average network size is 450 people
- Estimate: lawyers represent $2.6/450 = 0.58\%$ of the network
- Estimate: $0.0058 \cdot 290\,\text{million} = 1.7$ million lawyers in the U.S.

- On average, you know 0.25 people who were robbed last year
- Estimate: $\frac{0.25}{450} \cdot 290\,\text{million} = 160{,}000$ people robbed

Estimating group sizes: demonstration

- On average, you know 2.6 lawyers
- Assume average network size is 450 people
- Estimate: lawyers represent $2.6/450 = 0.58\%$ of the network
- Estimate: $0.0058 \cdot 290\,\text{million} = 1.7$ million lawyers in the U.S.

- On average, you know 0.25 people who were robbed last year
- Estimate: $\frac{0.25}{450} \cdot 290\,\text{million} = 160{,}000$ people robbed

Estimating group sizes: demonstration

- On average, you know 2.6 lawyers
- Assume average network size is 450 people
- Estimate: lawyers represent $2.6/450 = 0.58\%$ of the network
- Estimate: $0.0058 \cdot 290\,\text{million} = 1.7$ million lawyers in the U.S.

- On average, you know 0.25 people who were robbed last year
- Estimate: $\frac{0.25}{450} \cdot 290\,\text{million} = 160{,}000$ people robbed

Estimating group sizes: demonstration

- On average, you know 2.6 lawyers
- Assume average network size is 450 people
- Estimate: lawyers represent $2.6/450 = 0.58\%$ of the network
- Estimate: $0.0058 \cdot 290\,\text{million} = 1.7$ million lawyers in the U.S.

- On average, you know 0.25 people who were robbed last year
- Estimate: $\frac{0.25}{450} \cdot 290\,\text{million} = 160{,}000$ people robbed

Killworth, McCarty et al. surveys

- How many X's do you know?
- Stephanie, Jacqueline, Kimberly, Nicole, Christina, Jennifer
- Christopher, David, Anthony, Robert, James, Michael
- Twin, woman adopted kid in past year, gave birth in past year, widow(er) under 65
- Commercial pilot, gun dealer, postal worker, member of Jaycees, opened business in past year, American Indian
- Suicide in past year, died in auto accident, diabetic, kidney dialysis, AIDS, HIV-positive, rape victim, homicide victim, male in prison, homeless

Killworth, McCarty et al. surveys

- How many X's do you know?
- Stephanie, Jacqueline, Kimberly, Nicole, Christina, Jennifer
- Christopher, David, Anthony, Robert, James, Michael
- Twin, woman adopted kid in past year, gave birth in past year, widow(er) under 65
- Commercial pilot, gun dealer, postal worker, member of Jaycees, opened business in past year, American Indian
- Suicide in past year, died in auto accident, diabetic, kidney dialysis, AIDS, HIV-positive, rape victim, homicide victim, male in prison, homeless

Killworth, McCarty et al. surveys

- How many X's do you know?
- Stephanie, Jacqueline, Kimberly, Nicole, Christina, Jennifer
- Christopher, David, Anthony, Robert, James, Michael
- Twin, woman adopted kid in past year, gave birth in past year, widow(er) under 65
- Commercial pilot, gun dealer, postal worker, member of Jaycees, opened business in past year, American Indian
- Suicide in past year, died in auto accident, diabetic, kidney dialysis, AIDS, HIV-positive, rape victim, homicide victim, male in prison, homeless

Killworth, McCarty et al. surveys

- How many X's do you know?
- Stephanie, Jacqueline, Kimberly, Nicole, Christina, Jennifer
- Christopher, David, Anthony, Robert, James, Michael
- Twin, woman adopted kid in past year, gave birth in past year, widow(er) under 65
- Commercial pilot, gun dealer, postal worker, member of Jaycees, opened business in past year, American Indian
- Suicide in past year, died in auto accident, diabetic, kidney dialysis, AIDS, HIV-positive, rape victim, homicide victim, male in prison, homeless

Killworth, McCarty et al. surveys

- How many X's do you know?
- Stephanie, Jacqueline, Kimberly, Nicole, Christina, Jennifer
- Christopher, David, Anthony, Robert, James, Michael
- Twin, woman adopted kid in past year, gave birth in past year, widow(er) under 65
- Commercial pilot, gun dealer, postal worker, member of Jaycees, opened business in past year, American Indian
- Suicide in past year, died in auto accident, diabetic, kidney dialysis, AIDS, HIV-positive, rape victim, homicide victim, male in prison, homeless

Killworth, McCarty et al. surveys

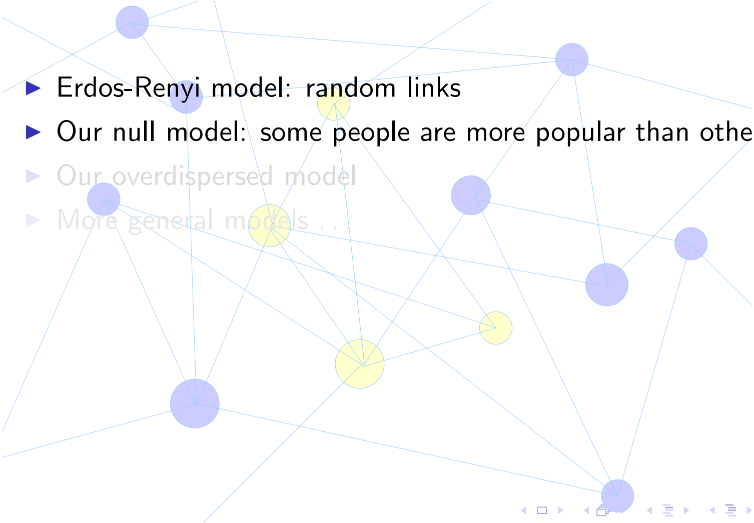
- How many X's do you know?
- Stephanie, Jacqueline, Kimberly, Nicole, Christina, Jennifer
- Christopher, David, Anthony, Robert, James, Michael
- Twin, woman adopted kid in past year, gave birth in past year, widow(er) under 65
- Commercial pilot, gun dealer, postal worker, member of Jaycees, opened business in past year, American Indian
- Suicide in past year, died in auto accident, diabetic, kidney dialysis, AIDS, HIV-positive, rape victim, homicide victim, male in prison, homeless

Models of social network data

- Erdos-Renyi model: random links
- Our null model: some people are more popular than others
- Our overdispersed model
- More general models . . .

Models of social network data

- Erdos-Renyi model: random links
- Our null model: some people are more popular than others
- Our overdispersed model
- More general models . . .



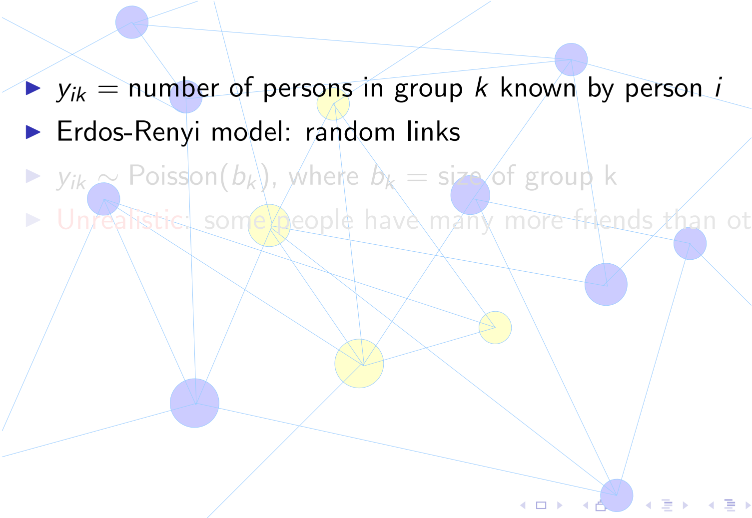
Models of social network data

- Erdos-Renyi model: random links
- Our null model: some people are more popular than others
- Our overdispersed model
- More general models . . .

Models of social network data

- Erdos-Renyi model: random links
- Our null model: some people are more popular than others
- Our overdispersed model
- More general models . . .

Erdos-Renyi model



- y_{ik} = number of persons in group k known by person i
- Erdos-Renyi model: random links
- $y_{ik} \sim \text{Poisson}(b_k)$, where b_k = size of group k
- Unrealistic: some people have many more friends than others

Erdos-Renyi model

- y_{ik} = number of persons in group k known by person i
- Erdos-Renyi model: random links
- $y_{ik} \sim \text{Poisson}(b_k)$, where b_k = size of group k
- Unrealistic: some people have many more friends than others

Erdos-Renyi model

- y_{ik} = number of persons in group k known by person i
- Erdos-Renyi model: random links
- $y_{ik} \sim \text{Poisson}(b_k)$, where b_k = size of group k
- Unrealistic: some people have many more friends than others

Our null model

- y_{ik} = number of persons in group k known by person i
- Our null model: some people are more popular than others
- $y_{ik} \sim \text{Poisson}(a_i b_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- Unrealistic: data are actually *overdispersed* (for example, do χ^2 test)

Our null model

- y_{ik} = number of persons in group k known by person i
- Our null model: some people are more popular than others
- $y_{ik} \sim \text{Poisson}(a_i b_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- Unrealistic: data are actually *overdispersed* (for example, do χ^2 test)

Our null model

- y_{ik} = number of persons in group k known by person i
- Our null model: some people are more popular than others
- $y_{ik} \sim \text{Poisson}(a_i b_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- Unrealistic: data are actually *overdispersed* (for example, do χ^2 test)

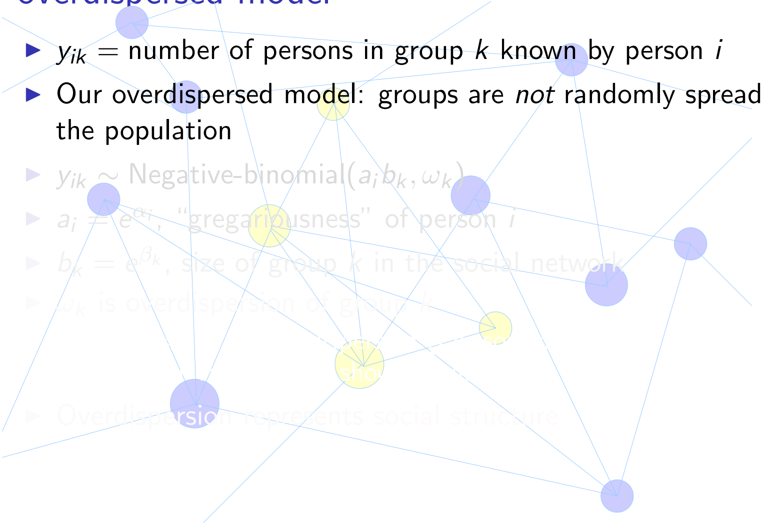
Our null model

- y_{ik} = number of persons in group k known by person i
- Our null model: some people are more popular than others
- $y_{ik} \sim \text{Poisson}(a_i b_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- Unrealistic: data are actually *overdispersed* (for example, do χ^2 test)

Our null model

- y_{ik} = number of persons in group k known by person i
- Our null model: some people are more popular than others
- $y_{ik} \sim \text{Poisson}(a_i b_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- Unrealistic: data are actually *overdispersed* (for example, do χ^2 test)

Our overdispersed model



- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim$ Negative-binomial$(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
- Overdispersion represents social structure

Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim \text{Negative-binomial}(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
- Overdispersion represents social structure

Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim \text{Negative-binomial}(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
  - $\omega_k = 1$ is no overdispersion (Poisson model). Higher values of ω_k show overdispersion
- Overdispersion represents social structure

Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim$ Negative-binomial$(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
  - $\omega_k = 1$ is no overdispersion (Poisson model)
  - Higher values of ω_k show overdispersion
- Overdispersion represents social structure

Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim$ Negative-binomial$(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
  - $\omega_k = 1$ is no overdispersion (Poisson model)
  - Higher values of ω_k show overdispersion
- Overdispersion represents social structure

Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim$ Negative-binomial$(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
  - $\omega_k = 1$ is no overdispersion (Poisson model)
  - Higher values of ω_k show overdispersion
- Overdispersion represents social structure

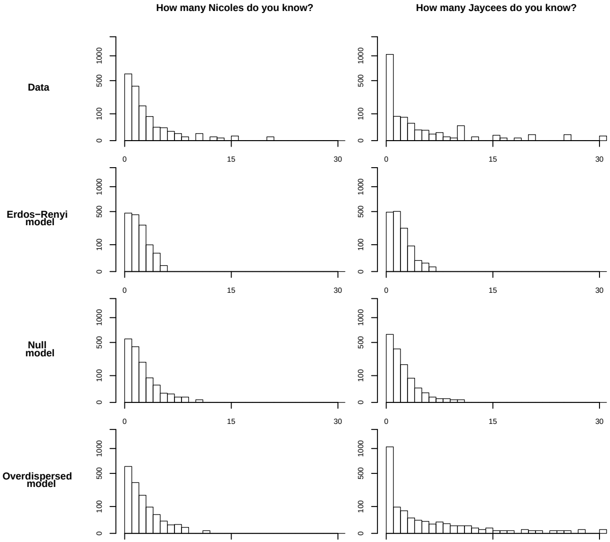
Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim$ Negative-binomial$(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
  - $\omega_k = 1$ is no overdispersion (Poisson model)
  - Higher values of ω_k show overdispersion
- Overdispersion represents social structure

Our overdispersed model

- y_{ik} = number of persons in group k known by person i
- Our overdispersed model: groups are *not* randomly spread in the population
- $y_{ik} \sim$ Negative-binomial$(a_i b_k, \omega_k)$
- $a_i = e^{\alpha_i}$, "gregariousness" of person i
- $b_k = e^{\beta_k}$, size of group k in the social network
- ω_k is overdispersion of group k
  - $\omega_k = 1$ is no overdispersion (Poisson model)
  - Higher values of ω_k show overdispersion
- Overdispersion represents social structure

Data, compared to simulations from 3 models



Bayesian inference

- Negative-binomial data model allowing overdispersion
- Hierarchical models for gregariousness, group-size, and overdispersion parameters
- $1370 + 32 + 32 + 4$ parameters to estimate
- Computation using the Gibbs/Metropolis sampler
- Adaptive (self-tuning) algorithm implemented using Jouni Kerman's Umacs function in R

Bayesian inference

- Negative-binomial data model allowing overdispersion
- Hierarchical models for gregariousness, group-size, and overdispersion parameters
- $1370 + 32 + 32 + 4$ parameters to estimate
- Computation using the Gibbs/Metropolis sampler
- Adaptive (self-tuning) algorithm implemented using Jouni Kerman’s Umacs function in R

Bayesian inference

- Negative-binomial data model allowing overdispersion
- Hierarchical models for gregariousness, group-size, and overdispersion parameters
- $1370 + 32 + 32 + 4$ parameters to estimate
- Computation using the Gibbs/Metropolis sampler
- Adaptive (self-tuning) algorithm implemented using Jouni Kerman's Umacs function in R

Bayesian inference

- Negative-binomial data model allowing overdispersion
- Hierarchical models for gregariousness, group-size, and overdispersion parameters
- $1370 + 32 + 32 + 4$ parameters to estimate
- Computation using the Gibbs/Metropolis sampler
- Adaptive (self-tuning) algorithm implemented using Jouni Kerman's Umacs function in R

Bayesian inference

- Negative-binomial data model allowing overdispersion
- Hierarchical models for gregariousness, group-size, and overdispersion parameters
- $1370 + 32 + 32 + 4$ parameters to estimate
- Computation using the Gibbs/Metropolis sampler
- Adaptive (self-tuning) algorithm implemented using Jouni Kerman's Umacs function in R

The overdispersed model

- data model: $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$, for $i = 1, \ldots, 1370$, $k = 1, \ldots, 32$
- prior dists
  - $\alpha_i \sim \text{N}(\mu_\alpha, \sigma^2_\alpha)$, for $i = 1, \ldots, 1370$
  - $\beta_k \sim \text{N}(\mu_\beta, \sigma^2_\beta)$, for $k = 1, \ldots, 32$
  - $\omega_k \sim \text{U}(1, 20)$, for $k = 1, \ldots, 32$
- hyperprior dist: $p(\mu_\alpha, \mu_\beta, \sigma_\alpha, \sigma_\beta) \propto 1$
- $1370 + 32 + 32 + 4$ parameters to estimate
- Nonidentifiability in $\alpha + \beta$ (to be discussed soon)

The overdispersed model

- data model: $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$, for $i = 1, \dots, 1370$, $k = 1, \dots, 32$
- prior dists
  - $\alpha_i \sim \text{N}(\mu_\alpha, \sigma^2_\alpha)$, for $i = 1, \dots, 1370$
  - $\beta_k \sim \text{N}(\mu_\beta, \sigma^2_\beta)$, for $k = 1, \dots, 32$
  - $\omega_k \sim \text{U}(1, 20)$, for $k = 1, \dots, 32$
- hyperprior dist: $p(\mu_\alpha, \mu_\beta, \sigma_\alpha, \sigma_\beta) \propto 1$
- $1370 + 32 + 32 + 4$ parameters to estimate
- Nonidentifiability in $\alpha + \beta$ (to be discussed soon)

The overdispersed model

- data model: $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$, for $i = 1, \ldots, 1370$, $k = 1, \ldots, 32$
- prior dists
  - $\alpha_i \sim \text{N}(\mu_\alpha, \sigma_\alpha^2)$, for $i = 1, \ldots, 1370$
  - $\beta_k \sim \text{N}(\mu_\beta, \sigma_\beta^2)$, for $k = 1, \ldots, 32$
  - $\omega_k \sim \text{U}(1, 20)$, for $k = 1, \ldots, 32$
- hyperprior dist: $p(\mu_\alpha, \mu_\beta, \sigma_\alpha, \sigma_\beta) \propto 1$
- $1370 + 32 + 32 + 4$ parameters to estimate
- Nonidentifiability in $\alpha + \beta$ (to be discussed soon)

The overdispersed model

- data model: $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$, for $i = 1, \ldots, 1370$, $k = 1, \ldots, 32$
- prior dists
  - $\alpha_i \sim \text{N}(\mu_\alpha, \sigma^2_\alpha)$, for $i = 1, \ldots, 1370$
  - $\beta_k \sim \text{N}(\mu_\beta, \sigma^2_\beta)$, for $k = 1, \ldots, 32$
  - $\omega_k \sim \text{U}(1, 20)$, for $k = 1, \ldots, 32$
- hyperprior dist: $p(\mu_\alpha, \mu_\beta, \sigma_\alpha, \sigma_\beta) \propto 1$
- $1370 + 32 + 32 + 4$ parameters to estimate
- Nonidentifiability in $\alpha + \beta$ (to be discussed soon)

The overdispersed model

- data model: $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$, for $i = 1, \ldots, 1370$, $k = 1, \ldots, 32$
- prior dists
  - $\alpha_i \sim \text{N}(\mu_\alpha, \sigma_\alpha^2)$, for $i = 1, \ldots, 1370$
  - $\beta_k \sim \text{N}(\mu_\beta, \sigma_\beta^2)$, for $k = 1, \ldots, 32$
  - $\omega_k \sim \text{U}(1, 20)$, for $k = 1, \ldots, 32$
- hyperprior dist: $p(\mu_\alpha, \mu_\beta, \sigma_\alpha, \sigma_\beta) \propto 1$
- $1370 + 32 + 32 + 4$ parameters to estimate
- Nonidentifiability in $\alpha + \beta$ (to be discussed soon)

Gibbs-Metropolis algorithm: updating α, β, ω

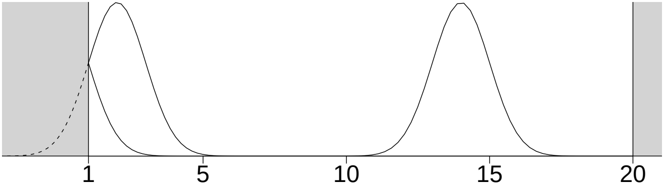
- For each i, update α_i using Metropolis with jumping dist. $\alpha_i^* \sim \mathrm{N}(\alpha_i^{(t-1)}, (\text{jumping scale of } \alpha_i)^2)$.
- For each k, update β_k using Metropolis with jumping dist. $\beta_k^* \sim \mathrm{N}(\beta_k^{(t-1)}, (\text{jumping scale of } \beta_k)^2)$.
- For each k, update ω_k using Metropolis with jumping dist. $\omega_k^* \sim \mathrm{N}(\omega_k^{(t-1)}, (\text{jumping scale of } \omega_k)^2)$. Reflect jumps off the edges:

Gibbs-Metropolis algorithm: updating α, β, ω

- For each i, update α_i using Metropolis with jumping dist. $\alpha_i^* \sim \mathrm{N}(\alpha_i^{(t-1)}, (\text{jumping scale of } \alpha_i)^2)$.
- For each k, update β_k using Metropolis with jumping dist. $\beta_k^* \sim \mathrm{N}(\beta_k^{(t-1)}, (\text{jumping scale of } \beta_k)^2)$.
- For each k, update ω_k using Metropolis with jumping dist. $\omega_k^* \sim \mathrm{N}(\omega_k^{(t-1)}, (\text{jumping scale of } \omega_k)^2)$. Reflect jumps off the edges:

Gibbs-Metropolis algorithm: updating α, β, ω

- For each i, update α_i using Metropolis with jumping dist. $\alpha_i^* \sim \mathrm{N}(\alpha_i^{(t-1)}, (\text{jumping scale of } \alpha_i)^2)$.
- For each k, update β_k using Metropolis with jumping dist. $\beta_k^* \sim \mathrm{N}(\beta_k^{(t-1)}, (\text{jumping scale of } \beta_k)^2)$.
- For each k, update ω_k using Metropolis with jumping dist. $\omega_k^* \sim \mathrm{N}(\omega_k^{(t-1)}, (\text{jumping scale of } \omega_k)^2)$.
  Reflect jumps off the edges:



Gibbs-Metropolis algorithm: updating hyperparameters

- Update $\mu_\alpha \sim \mathrm{N}\left(\frac{1}{n}\sum_{i=1}^{n}\alpha_i, \frac{1}{n}\sigma^2\right)$
- Update $\sigma_\alpha^2 \sim \text{Inv-}\chi^2\left(n-1, \frac{1}{n}\sum_{i=1}^{n}(\alpha_i - \mu_\alpha)^2\right)$
- Similarly with μ_β, σ_β

- Renormalize to identify the α's and β's ...

Gibbs-Metropolis algorithm: updating hyperparameters

- Update $\mu_\alpha \sim \text{N}\left(\frac{1}{n}\sum_{i=1}^{n} \alpha_i, \frac{1}{n}\sigma^2\right)$
- Update $\sigma_\alpha^2 \sim \text{Inv-}\chi^2\left(n-1, \frac{1}{n}\sum_{i=1}^{n} (\alpha_i - \mu_\alpha)^2\right)$
- Similarly with μ_β, σ_β
- Renormalize to identify the α's and β's ...

Gibbs-Metropolis algorithm: updating hyperparameters

- Update $\mu_\alpha \sim \mathrm{N}\left(\frac{1}{n}\sum_{i=1}^{n}\alpha_i, \frac{1}{n}\sigma^2\right)$
- Update $\sigma_\alpha^2 \sim \text{Inv-}\chi^2\left(n-1, \frac{1}{n}\sum_{i=1}^{n}(\alpha_i - \mu_\alpha)^2\right)$
- Similarly with μ_β, σ_β

- Renormalize to identify the α's and β's ...

Gibbs-Metropolis algorithm: updating hyperparameters

- Update $\mu_\alpha \sim \mathrm{N}\left(\frac{1}{n}\sum_{i=1}^{n} \alpha_i, \frac{1}{n}\sigma^2\right)$
- Update $\sigma_\alpha^2 \sim \text{Inv-}\chi^2\left(n-1, \frac{1}{n}\sum_{i=1}^{n}(\alpha_i - \mu_\alpha)^2\right)$
- Similarly with μ_β, σ_β

- Renormalize to identify the α's and β's ...

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:
  - Compute $C = \log\left(\sum_{k=1}^{12} e^{\beta_k}/0.069\right)$
  - Add C to all the α_i's and μ_α
  - Subtract C from all the β_k's and μ_β

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:
  - Compute $C = \log\left(\sum_{k=1}^{12} e^{\beta_k}/0.069\right)$
  - Add C to all the α_i's and μ_α
  - Subtract C from all the β_k's and μ_β

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:
  - Compute $C = \log\left(\sum_{k=1}^{12} e^{\beta_k}/0.069\right)$
  - Add C to all the α_i's and μ_α
  - Subtract C from all the β_k's and μ_β

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:
  - Compute $C = \log\left(\sum_{k=1}^{12} e^{\beta_k}/0.069\right)$
  - Add C to all the α_i's and μ_α
  - Subtract C from all the β_k's and μ_β

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:
  - Compute $C = \log\left(\sum_{k=1}^{12} e^{\beta_k}/0.069\right)$
  - Add C to all the α_i's and μ_α
  - Subtract C from all the β_k's and μ_β

Renormalizing the α_i's and β_k's

- Problem: α_i's and β_k's are not separately identified in the model, $y_{ik} \sim \text{Negative-binomial}(e^{\alpha_i+\beta_k}, \omega_k)$
- Possible solutions:
  - Choose a "baseline" value: set $\alpha_1 = 0$ (for example)
  - Renormalize a group of parameters: set $\sum_{i=1}^{n} \alpha_i = 0$
  - Anchor the prior distribution: set $\mu_\alpha = 0$
- Our solution: rescale so that the b_k's for the names (Nicole, Anthony, etc.) equal their proportion in the population:
  - Compute $C = \log\left(\sum_{k=1}^{12} e^{\beta_k}/0.069\right)$
  - Add C to all the α_i's and μ_α
  - Subtract C from all the β_k's and μ_β

Adaptive Metropolis jumping

- Parallel scalar updating of the components of α, β, ω
- Adapt each of $1370 + 32 + 32$ jumping scales to have $E(p_{\rm jump}) \approx 0.44$
- Save $p_{\rm jump}$ from each Metropolis step, then average them and rescale every 50 iterations:
  - Where avg $p_{\rm jump} > 0.44$, increase the jump scale
  - Where avg $p_{\rm jump} < 0.44$, decrease the jump scale
- After burn-in, stop adapting
- If we had vector jumps, we would adapt the scale so that $E(p_{\rm jump}) \approx 0.23$
- More effective adaptation uses avg. squared jumped distance

Adaptive Metropolis jumping

- Parallel scalar updating of the components of α, β, ω
- Adapt each of $1370 + 32 + 32$ jumping scales to have $E(p_{\text{jump}}) \approx 0.44$
- Save p_{jump} from each Metropolis step, then average them and rescale every 50 iterations:
  - Where avg $p_{\text{jump}} > 0.44$, increase the jump scale
  - Where avg $p_{\text{jump}} < 0.44$, decrease the jump scale
- After burn-in, stop adapting
- If we had vector jumps, we would adapt the scale so that $E(p_{\text{jump}}) \approx 0.23$
- More effective adaptation uses avg. squared jumped distance

Adaptive Metropolis jumping

- Parallel scalar updating of the components of α, β, ω
- Adapt each of $1370 + 32 + 32$ jumping scales to have $E(p_{\text{jump}}) \approx 0.44$
- Save p_{jump} from each Metropolis step, then average them and rescale every 50 iterations:
  - Where avg $p_{\text{jump}} > 0.44$, increase the jump scale
  - Where avg $p_{\text{jump}} < 0.44$, decrease the jump scale
- After burn-in, stop adapting
- If we had vector jumps, we would adapt the scale so that $E(p_{\text{jump}}) \approx 0.23$
- More effective adaptation uses avg. squared jumped distance

Adaptive Metropolis jumping

- Parallel scalar updating of the components of α, β, ω
- Adapt each of $1370 + 32 + 32$ jumping scales to have $E(p_{\rm jump}) \approx 0.44$
- Save $p_{\rm jump}$ from each Metropolis step, then average them and rescale every 50 iterations:
  - Where avg $p_{\rm jump} > 0.44$, increase the jump scale
  - Where avg $p_{\rm jump} < 0.44$, decrease the jump scale
- After burn-in, stop adapting
- If we had vector jumps, we would adapt the scale so that $E(p_{\rm jump}) \approx 0.23$
- More effective adaptation uses avg. squared jumped distance

Adaptive Metropolis jumping

- Parallel scalar updating of the components of α, β, ω
- Adapt each of $1370 + 32 + 32$ jumping scales to have $E(p_{\rm jump}) \approx 0.44$
- Save $p_{\rm jump}$ from each Metropolis step, then average them and rescale every 50 iterations:
  - Where avg $p_{\rm jump} > 0.44$, increase the jump scale
  - Where avg $p_{\rm jump} < 0.44$, decrease the jump scale
- After burn-in, stop adapting
- If we had vector jumps, we would adapt the scale so that $E(p_{\rm jump}) \approx 0.23$
- More effective adaptation uses avg. squared jumped distance

Adaptive Metropolis jumping

- Parallel scalar updating of the components of α, β, ω
- Adapt each of $1370 + 32 + 32$ jumping scales to have $E(p_{\rm jump}) \approx 0.44$
- Save $p_{\rm jump}$ from each Metropolis step, then average them and rescale every 50 iterations:
  - Where avg $p_{\rm jump} > 0.44$, increase the jump scale
  - Where avg $p_{\rm jump} < 0.44$, decrease the jump scale
- After burn-in, stop adapting
- If we had vector jumps, we would adapt the scale so that $E(p_{\rm jump}) \approx 0.23$
- More effective adaptation uses avg. squared jumped distance

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Computation in R

- BUGS was too slow (over 1400 parameters)
- Programming from scratch in R is awkward, buggy
- Instead, we use our general Gibbs/Metropolis programming environment
- Set up MCMC object
- Specify Gibbs updates
- Log-posterior density for Metropolis steps
- Bounds on overdispersion parameters $\omega \in [1, 20]$
- Renormalization step
- Result is a set of posterior simulations

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Setting up the MCMC object

```
network.1 <- mcmcEngine (network.data, network.init,
  update=network.update, n.iter=1000, n.chains=3)
network.update <- list(
  alpha = Metropolis (f.logpost.alpha),
  beta = Metropolis (f.logpost.beta),
  omega = Metropolis (f.logpost.omega,
    jump=Jump("omega.jump", lower=1.01, upper=20)),
  mu.alpha = Gibbs (mu.alpha.update),
  mu.beta = Gibbs (mu.beta.update),
  sigma.alpha = Gibbs (sigma.alpha.update),
  sigma.beta = Gibbs (sigma.beta.update),
  renorm.network)
```

Data and initial values

```
y <- as.matrix (read.dta ("social.dta"))
y <- y[1:50,]
network.data <- list (y=y, data.n=nrow(y),
  data.j=ncol(y))
network.init <- function(){
  alpha <- rnorm(data.n)
  beta <- rnorm(data.j)
  omega <- runif(data.j,1.01,20)
  mu.alpha <- rnorm(1)
  mu.beta <- rnorm(1)
  sigma.alpha <- runif(1)
  sigma.beta <- runif(1)}
```

Data and initial values

```
y <- as.matrix (read.dta ("social.dta"))
y <- y[1:50,]
network.data <- list (y=y, data.n=nrow(y),
  data.j=ncol(y))
network.init <- function(){
  alpha <- rnorm(data.n)
  beta <- rnorm(data.j)
  omega <- runif(data.j,1.01,20)
  mu.alpha <- rnorm(1)
  mu.beta <- rnorm(1)
  sigma.alpha <- runif(1)
  sigma.beta <- runif(1)}
```

Data and initial values

```
y <- as.matrix (read.dta ("social.dta"))
y <- y[1:50,]
network.data <- list (y=y, data.n=nrow(y),
  data.j=ncol(y))
network.init <- function(){
  alpha <- rnorm(data.n)
  beta <- rnorm(data.j)
  omega <- runif(data.j,1.01,20)
  mu.alpha <- rnorm(1)
  mu.beta <- rnorm(1)
  sigma.alpha <- runif(1)
  sigma.beta <- runif(1)}
```

Data and initial values

```
y <- as.matrix (read.dta ("social.dta"))
y <- y[1:50,]
network.data <- list (y=y, data.n=nrow(y),
  data.j=ncol(y))
network.init <- function(){
  alpha <- rnorm(data.n)
  beta <- rnorm(data.j)
  omega <- runif(data.j,1.01,20)
  mu.alpha <- rnorm(1)
  mu.beta <- rnorm(1)
  sigma.alpha <- runif(1)
  sigma.beta <- runif(1)}
```

Gibbs samplers for the hyperparameters

```
mu.alpha.update <- function()
  rnorm (1, mean(alpha), sigma.alpha/sqrt(data.n))
mu.beta.update <- function()
  rnorm (1, mean(beta), sigma.beta/sqrt(data.j))
sigma.alpha.update <- function()
  sqrt (sum((alpha-mu.alpha)^2)/rchisq(1, data.n-1))
sigma.beta.update <- function()
  sqrt (sum((beta-mu.beta)^2)/rchisq(1, data.j-1))
```

Gibbs samplers for the hyperparameters

```
mu.alpha.update <- function()
  rnorm (1, mean(alpha), sigma.alpha/sqrt(data.n))
mu.beta.update <- function()
  rnorm (1, mean(beta), sigma.beta/sqrt(data.j))
sigma.alpha.update <- function()
  sqrt (sum((alpha-mu.alpha)^2)/rchisq(1, data.n-1))
sigma.beta.update <- function()
  sqrt (sum((beta-mu.beta)^2)/rchisq(1, data.j-1))
```

Gibbs samplers for the hyperparameters

```
mu.alpha.update <- function()
  rnorm (1, mean(alpha), sigma.alpha/sqrt(data.n))
mu.beta.update <- function()
  rnorm (1, mean(beta), sigma.beta/sqrt(data.j))
sigma.alpha.update <- function()
  sqrt (sum((alpha-mu.alpha)^2)/rchisq(1, data.n-1))
sigma.beta.update <- function()
  sqrt (sum((beta-mu.beta)^2)/rchisq(1, data.j-1))
```

Gibbs samplers for the hyperparameters

```
mu.alpha.update <- function()
  rnorm (1, mean(alpha), sigma.alpha/sqrt(data.n))
mu.beta.update <- function()
  rnorm (1, mean(beta), sigma.beta/sqrt(data.j))
sigma.alpha.update <- function()
  sqrt (sum((alpha-mu.alpha)^2)/rchisq(1, data.n-1))
sigma.beta.update <- function()
  sqrt (sum((beta-mu.beta)^2)/rchisq(1, data.j-1))
```

Log-likelihood for each data point

```
f.loglik <- function (y, alpha, beta, omega, data.n){
  theta.mat <- exp(outer(alpha, beta, "+"))
  omega.mat <- outer(rep(0, data.n), omega, "+")
  dnbinom (y, theta.mat/(omega.mat-1), 1/omega.mat,
    log=T)}
```

Log-posterior density for each vector parameter

```
f.logpost.alpha <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  rowSums (loglik, na.rm=TRUE) +
    dnorm (alpha, mu.alpha, sigma.alpha, log=TRUE)}
f.logpost.beta <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  colSums (loglik, na.rm=TRUE) +
    dnorm (beta, mu.beta, sigma.beta, log=TRUE)}
f.logpost.omega <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  colSums (loglik, na.rm=T)}
```

Log-posterior density for each vector parameter

```
f.logpost.alpha <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  rowSums (loglik, na.rm=TRUE) +
    dnorm (alpha, mu.alpha, sigma.alpha, log=TRUE)}
f.logpost.beta <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  colSums (loglik, na.rm=TRUE) +
    dnorm (beta, mu.beta, sigma.beta, log=TRUE)}
f.logpost.omega <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  colSums (loglik, na.rm=T)}
```

Log-posterior density for each vector parameter

```
f.logpost.alpha <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  rowSums (loglik, na.rm=TRUE) +
    dnorm (alpha, mu.alpha, sigma.alpha, log=TRUE)}
f.logpost.beta <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  colSums (loglik, na.rm=TRUE) +
    dnorm (beta, mu.beta, sigma.beta, log=TRUE)}
f.logpost.omega <- function() {
  loglik <- f.loglik (y, alpha, beta, omega, data.n)
  colSums (loglik, na.rm=T)}
```

Bounded jumping for the ω_k's

Customized Metropolis jumping rule for the components of ω:

```
omega.jump <- function (omega, sigma) {
  reflect (rnorm (length(omega), omega, sigma),
    .lower, .upper)}
```

Renormalization of the α_i's and β_k's

```
renorm.network <- function() {
  const <- log (sum(exp(beta[1:12]))/0.069)
  alpha <- alpha + const
  mu.alpha <- mu.alpha + const
  beta <- beta - const
  mu.beta <- mu.beta - const}
```

Running MCMC and looking at the output

```
net <- run(network.1)
attach (as.rv (net))
```

Some output:

```
name       mean    sd    25%  50%  75%   Rhat
beta[1]    -5.1   0.1   (-5.4 -5.2 -5.1)  1.0
beta[2]    -6.4   0.1   (-6.9 -6.7 -6.5)  1.2
beta[3]    -6.1   0.1   (-6.5 -6.3 -6.2)  1.1
beta[4]    -7.0   0.2   (-7.6 -7.4 -7.1)  1.0
beta[5]    -5.1   0.1   (-5.4 -5.3 -5.2)  1.2
beta[6]    -5.6   0.2   (-6.1 -5.9 -5.8)  1.0
```

Running MCMC and looking at the output

```
net <- run(network.1)
attach (as.rv (net))
```

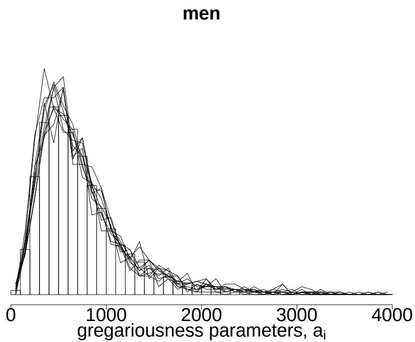
```
Some output:
name      mean    sd    25%  50%  75%   Rhat
beta[1]   -5.1   0.1   (-5.4 -5.2 -5.1)  1.0
beta[2]   -6.4   0.1   (-6.9 -6.7 -6.5)  1.2
beta[3]   -6.1   0.1   (-6.5 -6.3 -6.2)  1.1
beta[4]   -7.0   0.2   (-7.6 -7.4 -7.1)  1.0
beta[5]   -5.1   0.1   (-5.4 -5.3 -5.2)  1.2
beta[6]   -5.6   0.2   (-6.1 -5.9 -5.8)  1.0
```

Running MCMC and looking at the output

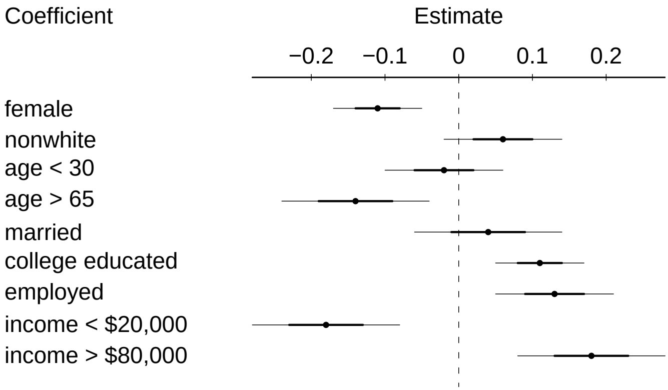
```
net <- run(network.1)
attach (as.rv (net))
```

```
Some output:
name      mean    sd    25%  50%  75%  Rhat
beta[1]   -5.1  0.1  (-5.4 -5.2 -5.1)  1.0
beta[2]   -6.4  0.1  (-6.9 -6.7 -6.5)  1.2
beta[3]   -6.1  0.1  (-6.5 -6.3 -6.2)  1.1
beta[4]   -7.0  0.2  (-7.6 -7.4 -7.1)  1.0
beta[5]   -5.1  0.1  (-5.4 -5.3 -5.2)  1.2
beta[6]   -5.6  0.2  (-6.1 -5.9 -5.8)  1.0
```

Estimated distributions of network sizes for men and women



Regression of log(gregariousness)



Parameter estimates for the 32 subpopulations

- Subpopulations
  - Names (Stephanie, Michael, etc.)
  - Other groups (pilots, diabetics, etc.)
- Parameters

Parameter estimates for the 32 subpopulations

- Subpopulations
  - Names (Stephanie, Michael, etc.)
  - Other groups (pilots, diabetics, etc.)
- Parameters
  - Proportion of the social network, e^{β_k}
  - Overdispersion, ω_k

Parameter estimates for the 32 subpopulations

- Subpopulations
  - Names (Stephanie, Michael, etc.)
  - Other groups (pilots, diabetics, etc.)
- Parameters
  - Proportion of the social network, e^{β_k}
  - Overdispersion, ω_k

Parameter estimates for the 32 subpopulations

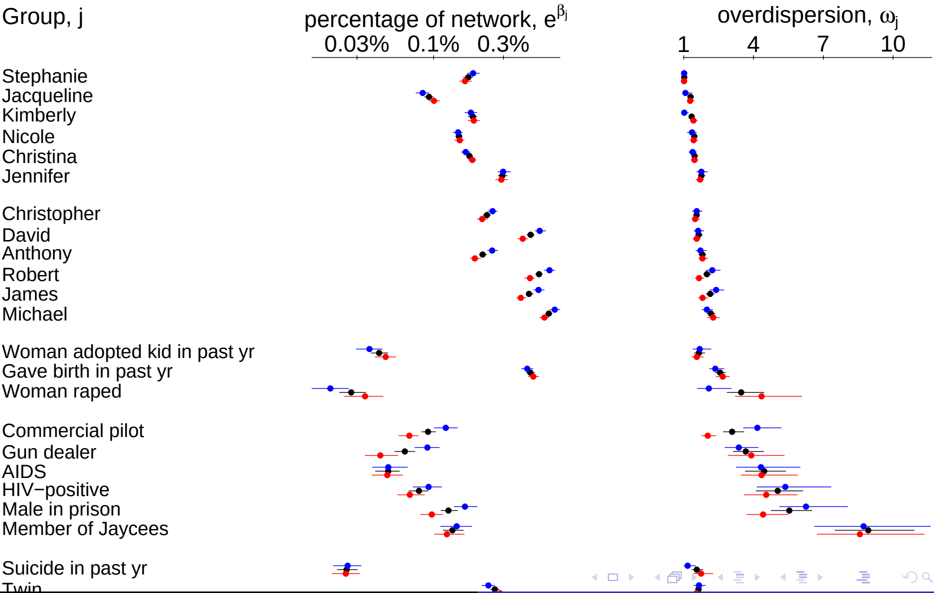
- Subpopulations
  - Names (Stephanie, Michael, etc.)
  - Other groups (pilots, diabetics, etc.)
- Parameters
  - Proportion of the social network, e^{β_k}
  - Overdispersion, ω_k

Parameter estimates for the 32 subpopulations

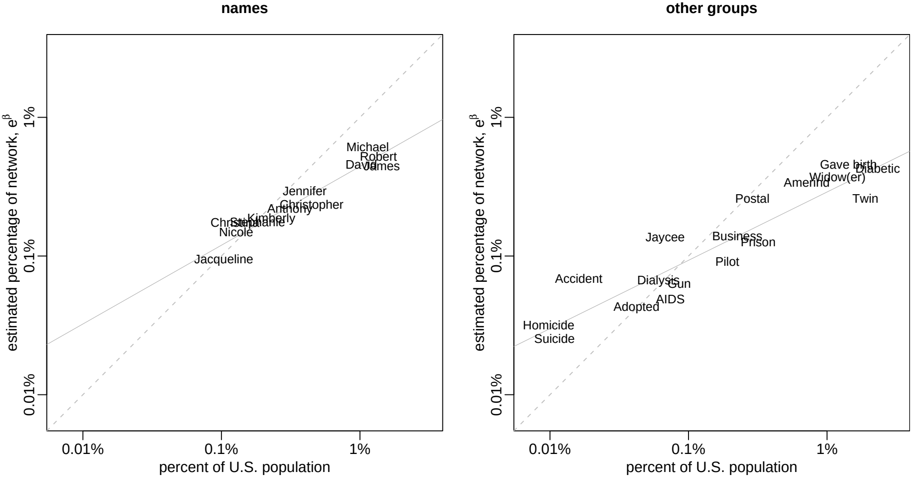
- Subpopulations
  - Names (Stephanie, Michael, etc.)
  - Other groups (pilots, diabetics, etc.)
- Parameters
  - Proportion of the social network, e^{β_k}
  - Overdispersion, ω_k

Parameter estimates for the 32 subpopulations

- Subpopulations
  - Names (Stephanie, Michael, etc.)
  - Other groups (pilots, diabetics, etc.)
- Parameters
  - Proportion of the social network, e^{β_k}
  - Overdispersion, ω_k


Group, j
percentage of network, e^{β_j}
0.03%
0.1%
0.3%
overdispersion, ω_j
1
4
7
10
Stephanie
Jacqueline
Kimberly
Nicole
Christina
Jennifer
Christopher
David
Anthony
Robert
James
Michael
Woman adopted kid in past yr
Gave birth in past yr
Woman raped
Commercial pilot
Gun dealer
AIDS
HIV−positive
Male in prison
Member of Jaycees
Suicide in past yr
Twin

Comparing estimated and actual group sizes



Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
- Explanations
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accidents, etc.) are over-recalled. Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations

- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
  - Difficulty recalling all the Michaels you know
  - Salience of rare events in memory
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
  - Difficulty recalling all the Michaels you know
  - Salience of rare events in memory
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
  - Difficulty recalling all the Michaels you know
  - Salience of rare events in memory
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
  - Difficulty recalling all the Michaels you know
  - Salience of rare events in memory
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

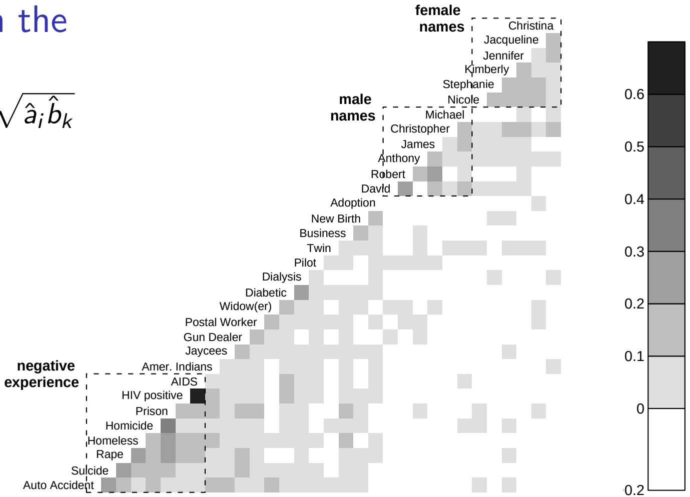
- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
  - Difficulty recalling all the Michaels you know
  - Salience of rare events in memory
- Recall Nicole and Anthony from the demo!

Comparing estimated and actual group sizes

- Names
  - Rare names (Stephanie, Nicole, etc.) fit their population frequencies
  - Common names (Michael, Robert, etc.) are underrepresented in the friendship network
- Other groups
  - Rare groups (homicide, accident, etc.) are over-recalled
  - Common groups (new mothers, diabetics, etc.) are under-recalled
- Explanations
  - Difficulty recalling all the Michaels you know
  - Salience of rare events in memory
- Recall Nicole and Anthony from the demo!

Correlations in the residuals

$$r_{ik} = \sqrt{y_{ik}} - \sqrt{\hat{a}_i \hat{b}_k}$$



Confidence building

- Posterior predictive checking: compare data to simulated replications from the model
  - Model fit is good, not perfect
  - Consistent patterns with names compared to other groups
  - Many fewer 9's and more 10's in data than predicted by the model
- Checking parameter estimates under fake-data simulation

Confidence building

- Posterior predictive checking: compare data to simulated replications from the model
  - Model fit is good, not perfect
  - Consistent patterns with names compared to other groups
  - Many fewer 9's and more 10's in data than predicted by the model
- Checking parameter estimates under fake-data simulation

Confidence building

- Posterior predictive checking: compare data to simulated replications from the model
  - Model fit is good, not perfect
  - Consistent patterns with names compared to other groups
  - Many fewer 9's and more 10's in data than predicted by the model
- Checking parameter estimates under fake-data simulation

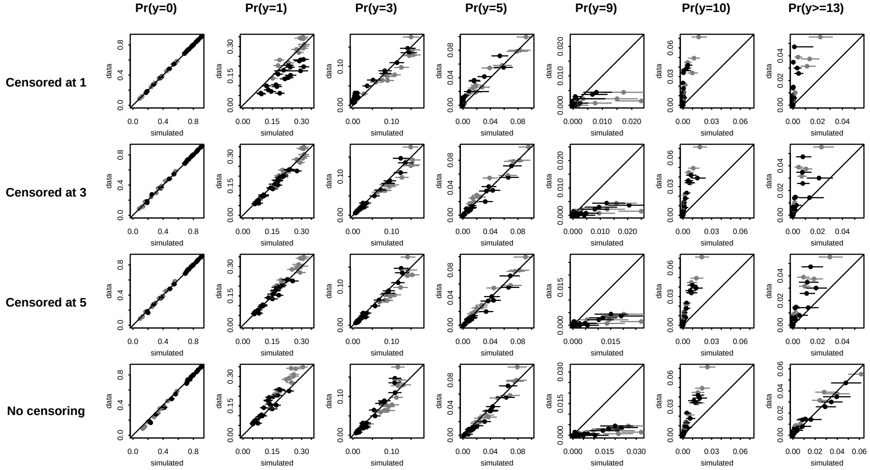
Confidence building

- Posterior predictive checking: compare data to simulated replications from the model
  - Model fit is good, not perfect
  - Consistent patterns with names compared to other groups
  - Many fewer 9's and more 10's in data than predicted by the model
- Checking parameter estimates under fake-data simulation

Confidence building

- Posterior predictive checking: compare data to simulated replications from the model
  - Model fit is good, not perfect
  - Consistent patterns with names compared to other groups
  - Many fewer 9's and more 10's in data than predicted by the model
- Checking parameter estimates under fake-data simulation

Actual vs. simulated proportions of $y = 0, 1, \ldots$



Do you know 0, 1, 2, or 3 or more Nicoles?

- Censored-data model
- $y_{ik} = 0, 1, 2, \text{ or } \geq 3$
- Use negative-binomial likelihood function: $\Pr(y=0)$, $\Pr(y=1)$, $\Pr(y=2)$, $1 - \Pr(y=0) - \Pr(y=1) - \Pr(y=2)$
- Gibbs-Metropolis algorithm is otherwise unchanged
- Check with our data: parameter estimates are similar but problems with model fit for high values of y

Do you know 0, 1, 2, or 3 or more Nicoles?

- Censored-data model
- $y_{ik} = 0, 1, 2$, or ≥ 3
- Use negative-binomial likelihood function: $\Pr(y=0)$, $\Pr(y=1)$, $\Pr(y=2)$, $1 - \Pr(y=0) - \Pr(y=1) - \Pr(y=2)$
- Gibbs-Metropolis algorithm is otherwise unchanged
- Check with our data: parameter estimates are similar but problems with model fit for high values of y

Do you know 0, 1, 2, or 3 or more Nicoles?

- Censored-data model
- $y_{ik} = 0, 1, 2, \text{ or } \geq 3$
- Use negative-binomial likelihood function:
  $\Pr(y\!=\!0), \Pr(y\!=\!1), \Pr(y\!=\!2),$
  $1 - \Pr(y\!=\!0) - \Pr(y\!=\!1) - \Pr(y\!=\!2)$
- Gibbs-Metropolis algorithm is otherwise unchanged
- Check with our data: parameter estimates are similar but problems with model fit for high values of y

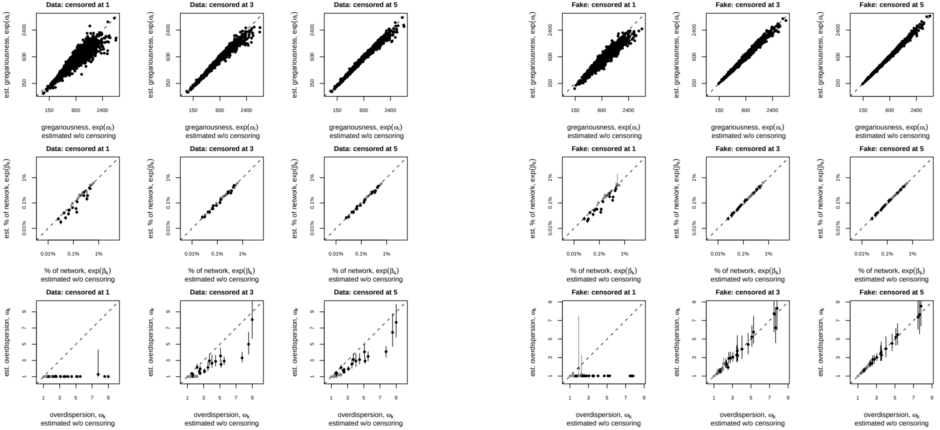
Do you know 0, 1, 2, or 3 or more Nicoles?

- Censored-data model
- $y_{ik} = 0, 1, 2, \text{ or } \geq 3$
- Use negative-binomial likelihood function:
  $\Pr(y=0)$, $\Pr(y=1)$, $\Pr(y=2)$,
  $1 - \Pr(y=0) - \Pr(y=1) - \Pr(y=2)$
- Gibbs-Metropolis algorithm is otherwise unchanged
- Check with our data: parameter estimates are similar but problems with model fit for high values of y

Do you know 0, 1, 2, or 3 or more Nicoles?

- Censored-data model
- $y_{ik} = 0$, 1, 2, or ≥ 3
- Use negative-binomial likelihood function:
  $\Pr(y\!=\!0)$, $\Pr(y\!=\!1)$, $\Pr(y\!=\!2)$,
  $1 - \Pr(y\!=\!0) - \Pr(y\!=\!1) - \Pr(y\!=\!2)$
- Gibbs-Metropolis algorithm is otherwise unchanged
- Check with our data: parameter estimates are similar but problems with model fit for high values of y

Evaluation of inferences using fake data



Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds); 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Running the demo

- How many Nicoles, Anthonys, lawyers, people robbed?
- Real-time data analysis
  - Entering in the data: 20 minutes
  - Running the program: 500 iterations (40 seconds), 1000 iterations (80 seconds)
  - Real-time debugging: 15 minutes!
  - Altering the presentation: 15 minutes!
- Results for social network sizes, α
- Results for group sizes, β
- Results for overdispersions, ω

Results of the demo

- Social network sizes, α
  - Mean network size estimated at 370 ± 20
  - We don't really believe this precision!
  - Implicit hierarchical model

Results of the demo

- Social network sizes, α
  - Mean network size estimated at 370 ± 20
  - We don't really believe this precision!
  - Implicit hierarchical model

Results of the demo

- Social network sizes, α
  - Mean network size estimated at 370 ± 20
  - We don't really believe this precision!
  - Implicit hierarchical model

Results of the demo

- Social network sizes, α
  - Mean network size estimated at 370 ± 20
  - We don't really believe this precision!
  - Implicit hierarchical model

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Group sizes, β
  - Nicole: 0.17% of the social network
  - Anthony: 0.27% of the social network
  - Lawyers: 0.90% of the social network
  - Robbed last year: 0.20% of the social network
- Scale-up
  - Nicole: 500,000
  - Anthony: 800,000
  - Lawyers: 2.6 million
  - Robbed last year: 200,000

Results of the demo

- Overdispersions, ω
  - Nicole: 1.1 ± 0.1
  - Anthony: 1.2 ± 0.1
  - Lawyers: 4.2 ± 0.9
  - Robbed last year: 1.3 ± 0.3

Results of the demo

- Overdispersions, ω
  - Nicole: 1.1 ± 0.1
  - Anthony: 1.2 ± 0.1
  - Lawyers: 4.2 ± 0.9
  - Robbed last year: 1.3 ± 0.3

Results of the demo

- Overdispersions, ω
  - Nicole: 1.1 ± 0.1
  - Anthony: 1.2 ± 0.1
  - Lawyers: 4.2 ± 0.9
  - Robbed last year: 1.3 ± 0.3

Results of the demo

- Overdispersions, ω
  - Nicole: 1.1 ± 0.1
  - Anthony: 1.2 ± 0.1
  - Lawyers: 4.2 ± 0.9
  - Robbed last year: 1.3 ± 0.3

Results of the demo

- Overdispersions, ω
  - Nicole: 1.1 ± 0.1
  - Anthony: 1.2 ± 0.1
  - Lawyers: 4.2 ± 0.9
  - Robbed last year: 1.3 ± 0.3

Conclusions

- Bayesian data analysis
- What we learned about social networks
- Advantages of "How many X's" surveys
- Plan of future research

Conclusions

- Bayesian data analysis
- What we learned about social networks
- Advantages of "How many X's" surveys
- Plan of future research

Conclusions

- Bayesian data analysis
- What we learned about social networks
- Advantages of "How many X's" surveys
- Plan of future research

Conclusions

- Bayesian data analysis
- What we learned about social networks
- Advantages of "How many X's" surveys
- Plan of future research

Conclusions

- Bayesian data analysis
- What we learned about social networks
- Advantages of "How many X's" surveys
- Plan of future research

Bayesian data analysis

- Model-building motivated by failures of simpler models
- Checking model by comparing data to predictive replications
- Checking computer program by checking inferences from fake data
- Computation using automated Metropolis algorithm
- Inferences summarized graphically . . .

Bayesian data analysis

- Model-building motivated by failures of simpler models
- Checking model by comparing data to predictive replications
- Checking computer program by checking inferences from fake data
- Computation using automated Metropolis algorithm
- Inferences summarized graphically . . .

Bayesian data analysis

- Model-building motivated by failures of simpler models
- Checking model by comparing data to predictive replications
- Checking computer program by checking inferences from fake data
- Computation using automated Metropolis algorithm
- Inferences summarized graphically . . .

Bayesian data analysis

- Model-building motivated by failures of simpler models
- Checking model by comparing data to predictive replications
- Checking computer program by checking inferences from fake data
- Computation using automated Metropolis algorithm
- Inferences summarized graphically . . .

Bayesian data analysis

- Model-building motivated by failures of simpler models
- Checking model by comparing data to predictive replications
- Checking computer program by checking inferences from fake data
- Computation using automated Metropolis algorithm
- Inferences summarized graphically ...

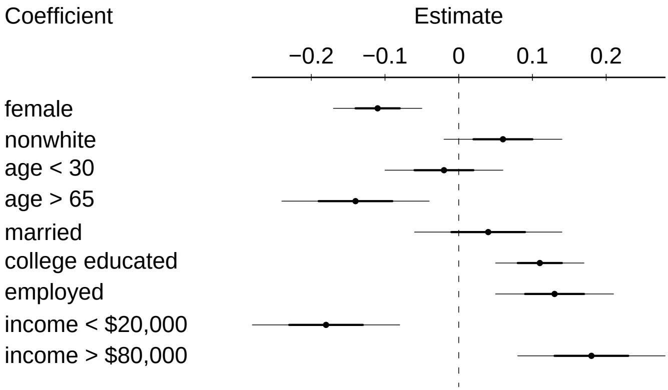
Bayesian data analysis

- Model-building motivated by failures of simpler models
- Checking model by comparing data to predictive replications
- Checking computer program by checking inferences from fake data
- Computation using automated Metropolis algorithm
- Inferences summarized graphically . . .

Regression of log(gregariousness): as a table

Coefficient	Estimate (s.e.)
female	−0.11 (0.03)
nonwhite	0.06 (0.04)
age < 30	−0.02 (0.04)
age > 65	−0.14 (0.05)
married	0.04 (0.05)
college educated	0.11 (0.03)
employed	0.13 (0.04)
income $< \$20,000$	−0.18 (0.05)
income $> \$80,000$	0.18 (0.05)

Regression of log(gregariousness): as a graph



What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion

What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion
  - Names are roughly uniformly distributed
  - Some other groups show more structure
  - Potential for regression models (with geographic and social predictors)

What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion
  - Names are roughly uniformly distributed
  - Some other groups show more structure
  - Potential for regression models (with geographic and social predictors)

What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion
  - Names are roughly uniformly distributed
  - Some other groups show more structure
  - Potential for regression models (with geographic and social predictors)

What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion
  - Names are roughly uniformly distributed
  - Some other groups show more structure
  - Potential for regression models (with geographic and social predictors)

What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion
  - Names are roughly uniformly distributed
  - Some other groups show more structure
  - Potential for regression models (with geographic and social predictors)

What have we learned about social networks

- Network size
  - On average, people know about 750 people
  - Distribution is similar for men and women
- Overdispersion
  - Names are roughly uniformly distributed
  - Some other groups show more structure
  - Potential for regression models (with geographic and social predictors)

Learning from “How many X’s” surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, . . .)
- Difficulty with recall
- Potential design using partial information:

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, . . .)
- Difficulty with recall
- Potential design using partial information:

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:
  - Do you know any Nicoles?
    Do you know 0, 1, 2, or 3 or more Nicoles?

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:
  - Do you know any Nicoles?
  - Do you know 0, 1, 2, or 3 or more Nicoles?

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:
  - Do you know any Nicoles?
  - Do you know 0, 1, 2, or 3 or more Nicoles?

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:
  - Do you know any Nicoles?
  - Do you know 0, 1, 2, or 3 or more Nicoles?

Learning from "How many X's" surveys

- Network info from a non-network sample
- We can even learn about small groups, less than 0.3% of population
- Implicit survey of $1500 \times 750 = 1$ *million* people!
- Characterising people by how they are perceived
- Potentially useful for small or hard-to-reach groups (prisoners, ...)
- Difficulty with recall
- Potential design using partial information:
  - Do you know any Nicoles?
  - Do you know 0, 1, 2, or 3 or more Nicoles?

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with lots of parameters
- Conduct new survey (GSS module, possibly NES also)

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)

Our research plan

- Design and analysis of “How many X’s” surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)
  - Goals: estimating overdispersion of subpopulations, regression models of # known and individual characteristics and attitudes Measuring and understanding social and political polarization Leraning about individuals and groups

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)
  - Goals: estimating overdispersion of subpopulations, regression models of # known and individual characteristics and attitudes
  - Measuring and understanding social and political polarization
  - Learning about individuals and groups

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or ...?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)
  - Goals: estimating overdispersion of subpopulations, regression models of # known and individual characteristics and attitudes
  - Measuring and understanding social and political polarization
  - Leraning about individuals and groups

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)
  - Goals: estimating overdispersion of subpopulations, regression models of # known and individual characteristics and attitudes
  - Measuring and understanding social and political polarization
  - Leraning about individuals and groups

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)
  - Goals: estimating overdispersion of subpopulations, regression models of # known and individual characteristics and attitudes
  - Measuring and understanding social and political polarization
  - Leraning about individuals and groups

Our research plan

- Design and analysis of "How many X's" surveys
  - Ask about 0/1+, or 0/1/2+, or . . . ?
  - Use rare names to normalize?
  - Efficient estimation given fixed respondent time
  - Hierarchical regression models with `lots` of parameters
- Conduct new survey (GSS module, possibly NES also)
  - Goals: estimating overdispersion of subpopulations, regression models of # known and individual characteristics and attitudes
  - Measuring and understanding social and political polarization
  - Leraning about individuals and groups