SOC542 Statistical Methods in Sociology II

Missing Data & Model Checking and Robustness

Thomas Davidson

Rutgers University

March 7, 2022

Plan

- Missing data
- Model checking
- Model robustness

Missing data

What is missing data?

- Missing data occurs when we do not have an record of any data for one or more variables for an observation.
- This can occur for a variety of reasons including
  - Skip patterns
  - Survey non-response
  - Survey error
  - Data entry errors
  - Data handling errors
- The severity of the problem depends on why the data are missing and the amount of missingness

Missing data

Missing completely at random (MCAR)

- **Missing Completely at Random (MCAR)**
  - Probability x_i is missing is constant across all observations
- Discarding missing cases does not result in any bias

Missing data

Missing at random (MAR)

- **Missing at Random (MAR)**
  - Probability x_i is missing depends on observed variables.
- Discarding missing cases does not result in any bias if predictors of missingness are adjusted for.

Missing data

Missing not at random

- Missingness that depends on the missing value is considered **Missing not at Random (MNAR)**
  - e.g. Higher income respondents less likely to report income

Missing data

Simulating missingness

- We can use simulations to better understand the effects to different kinds of missingness
- Consider the following population model

$$z = N(0,1)$$

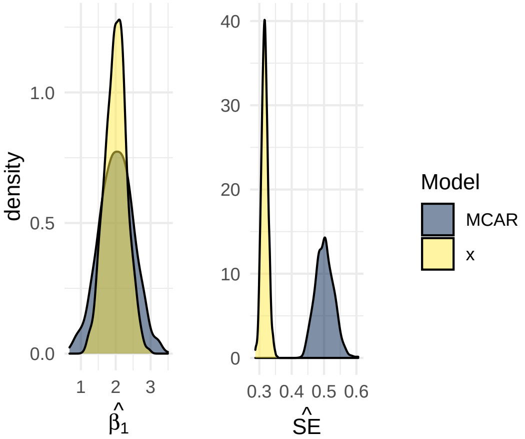
$$x = N(0,1)$$

$$y = 2x - 2z + N(0,10)$$

- We can vary the kind of missingess in x and analyze how it effects $\hat{\beta}_1$ and $\hat{\sigma}$.

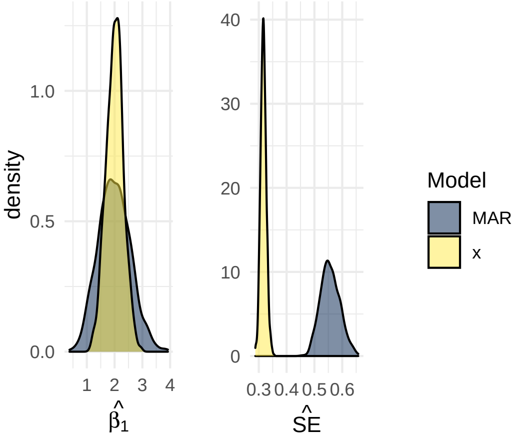
Missing data

MCAR, $p(Missing) = 0.6$



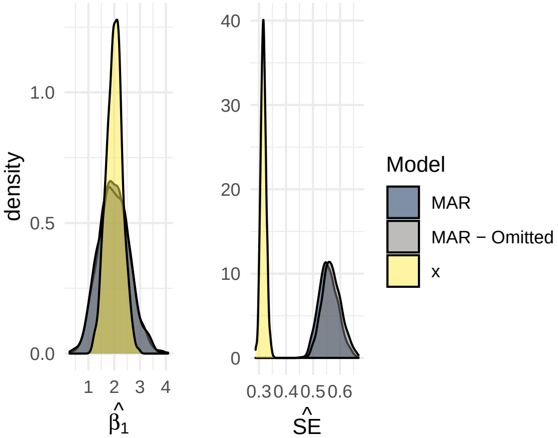
Missing data

MAR, $p(Missing) = 0.8$ if $z > 0$



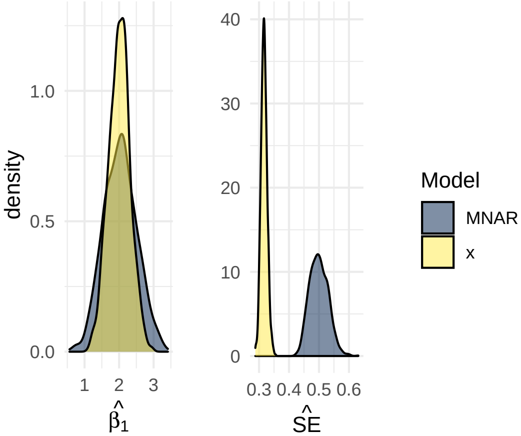
Missing data

MAR, *z* omitted



Missing data

MNAR, $p(Missing) = 0.8$ if $x > 0$



Missing data

Missingness in practice

- It is often difficult to determine why data are missing
    - Knowledge of domain and data generation process helpful
- MCAR is a very strong assumption
- MAR is more reasonable, but it is difficult to determine whether missingness is due to an unobserved factor
    - Including more predictors in a model helps to reduce concerns

Missing data

Addressing missingness

- Three approaches
  1. Delete missing cases
  2. *Simple* imputation
  3. *Multiple* imputation

Missing data

Deleting missing data

- **Complete-case analysis / listwise deletion**
  - Delete all rows where y or x or z is missing
- If missingess is not MCAR then results could be biased
- If many predictors, sample size can reduce substantially

Missing data

Deleting missing data

- **Available-case analysis / pairwise deletion**
  - Make comparisons where data are available
- In the previous example, one might use the following model to get an estimate of the effect of z on y

$$y = \beta_0 + \beta_1 z + u$$

- The variable x is ignored, otherwise we drop cases where x is missing.
- One might also estimate the model including x and compare the results.
- Like listwise deletion, bias can occur if systematic differences between missing and non-missing cases.

Missing data

Simple imputation: Guessing the mean

- Instead of removing data, we can try to "guess" the missing values.
- A good guess is the mean of the observed data, **mean imputation**.
- But this can distort the distribution and underestimate the standard deviation.

Missing data

Simple imputation

```
X <- rnorm(N)
print(mean(X))
```

```
## [1] 0.009998062
```

```
print(sd(X))
```

```
## [1] 1.020109
```

```
X.m <- ifelse(rbinom(N, 1, 1-0.2), X, NA)
X.g <- ifelse(!is.na(X.m), X, mean(X.m[!is.na(X.m)]))
print(mean(X.g))
```

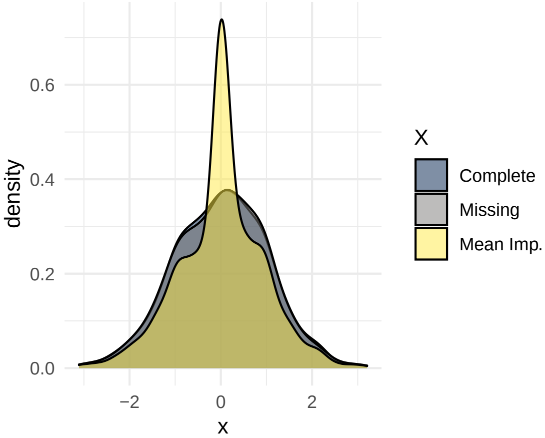
```
## [1] 0.01318459
```

```
print(sd(X.g))
```

```
## [1] 0.922013
```

Missing data

Simple imputation



Missing data

Simple imputation: Random imputation

- We can address some of the limitations of mean imputation by using the full distribution of the observed variable. This is known as **random imputation**
- However, the imputed values will not necessarily reflect the underlying association between variables and we do not make use of other information.

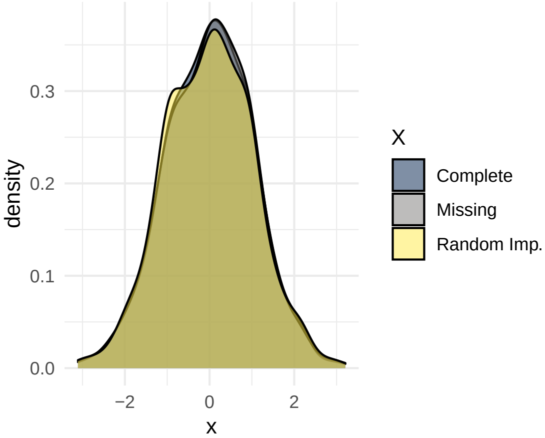
Missing data

Simple imputation: Random imputation

```
missing <- is.na(X.m)
X.g2 <- X.m
X.g2[missing] <- sample(X.m[!is.na(X.m)],
                        length(X.m[missing]),
                        replace = TRUE)
```

Missing data

Simple imputation: Random imputation



Missing data

Simple imputation: Prediction with regression

- We can improve our imputation model by making use of additional information in other variables.
  - For example, we could estimate a model to predict x using other covariates and then use these predictions in place of missing vaulues

Missing data

Simple imputation: Prediction with regression

```
N <- 1000
v <- rnorm(N)
w <- rnorm(N)
x <- 0.5*v + 0.5*w + rnorm(N)
y <- x - 2*w + rnorm(N)

x.m <- ifelse(rbinom(N, 1, 0.4), x, NA) # MCAR

imp.model <- lm(x.m ~ v + w)
```

Missing data

Simple imputation: Prediction with regression

```
preds <- predict(imp.model,
                 newdata = as.data.frame(cbind(v, w)))

x.imp <- x.m
for (i in 1:length(x.m)) {
    if (is.na(x.m[i])) {
        x.imp[i] <- preds[i]
    }
}
```

Missing data

Simple imputation: Prediction with regression

	Complete	Listwise	Mean	Random	Predicted
(Intercept)	-0.016	-0.043	-0.011	-0.007	-0.008
	(0.032)	(0.052)	(0.043)	(0.046)	(0.040)
x	0.994	1.034	0.915	0.309	0.998
	(0.030)	(0.048)	(0.061)	(0.040)	(0.052)
w	-2.007	-2.082	-1.703	-1.581	-1.968
	(0.036)	(0.056)	(0.045)	(0.048)	(0.047)
Num.Obs.	1000	390	1000	1000	1000
R2	0.765	0.790	0.591	0.527	0.635
R2 Adj.	0.764	0.788	0.591	0.526	0.634

Missing data

Simple imputation

- Limitations
  - Each process becomes cumbersome if we have multiple variables with missing data and observations with more than one variable missing.
    - e.g. How do we predict x if we are missing other variables?
  - These approaches are *deterministic*, failing to take into account the uncertainity in the imputations.

Missing data

Multiple imputation

- **Multiple imputation (MI)** methods address both issues:
  - MI models use observed data to predict missing values.
  - Multiple missing variable can be imputed simultaneously.
  - Prediction uncertainity can be incorporated into the estimates.

Missing data

Multiple imputation: Algorithms

- Multiple imputation algorithms work by using existing data to predict missing values across the entire dataset
- Generally, these algorithms use iterative procedures, predicting a subset of the missing values at a time
- These algorithms converge when the distributions of the predicted datasets look like the distributions in the original data
- Often these algorithms use Bayesian techniques such as MCMC sampling[1]

[1] See the mi command in Stata for example.

Missing data

Multiple imputation: MI in R

- There are a number of different MI packages available in R
- Two commonly used packages are
  - MICE (Multivariate Imputation via Chained Equations)
  - Amelia
    - Particularly useful for panel data

Missing data

Multiple imputation: Pooling

- MI algorithms can produce M imputed datasets.
  - In each case, we can compute an estimate, $\hat{\beta}_{1m}$ and a standard error $\hat{SE}_m$.
- The overall estimate is an average over the M datasets, known as a **pooled** estimate:[2]

$$\hat{\beta} = \frac{1}{M} \sum_{m=1}^{M} \hat{\beta}_{1m}$$

[2] See GHV p. 326 for the formula for the standard error of the pooled estimate.

Missing data

Multiple imputation with MICE

```r
x <- x.m
data <- cbind(y, v, w, x)

M <- mice(data, m=10, method = "pmm",
          seed=08901,
          printFlag = FALSE)
```

Missing data

Multiple imputation with MICE

```r
# Impute using m=1
simple.imp <- complete(M,1)
mi.s <- lm(y ~ x + w, data = simple.imp)

# Pool over all M
fits <- with(M, lm(y ~ x + w))
mi.M <- pool(fits)
```

Missing data

Multiple imputation with MICE

	Complete	Predicted	MI m	MI M
(Intercept)	-0.016	-0.008	-0.029	-0.029
	(0.032)	(0.040)	(0.033)	(0.044)
x	0.994	0.998	1.007	1.021
	(0.030)	(0.052)	(0.031)	(0.038)
w	-2.007	-1.968	-2.002	-2.030
	(0.036)	(0.047)	(0.037)	(0.050)
Num.Obs.	1000	1000	1000	1000
Num.Imp.				10
R2	0.765	0.635	0.756	0.758
R2 Adj.	0.764	0.634	0.755	0.758

Model checking

Diagnostics

- We have already covered several different diagnostics for model checking
  - Residuals and standard error of the residuals
  - Predicted values
  - R^2 and adjusted R^2
  - Standard errors on coefficients and p-values
  - F-statistic

Model checking

Outliers

- Outliers are extreme data points that deviate from the distribution of other values
  - Scatterplots of raw data and residual can be helpful for identifying these
- An outlier has **leverage** if the addition of the observation results in a change in the slope of the regression line
- Such cases are considered *influential* if they result in substantial changes to the regression results
  - e.g. Differences in statistical sigificance, sign, magnitude

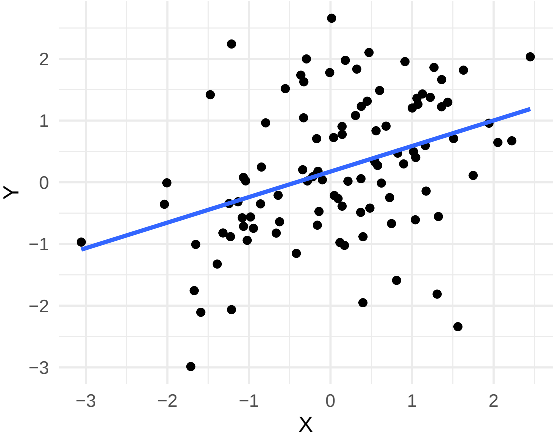
Model checking

Outliers

```
N <- 100
X <- rnorm(N)
Y <- 0.5*X + rnorm(N)
```

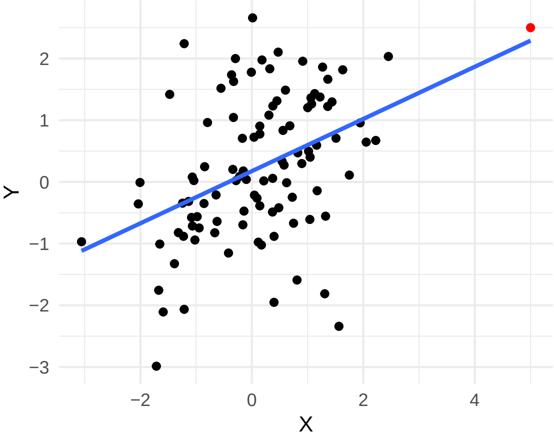
Model checking

Outliers



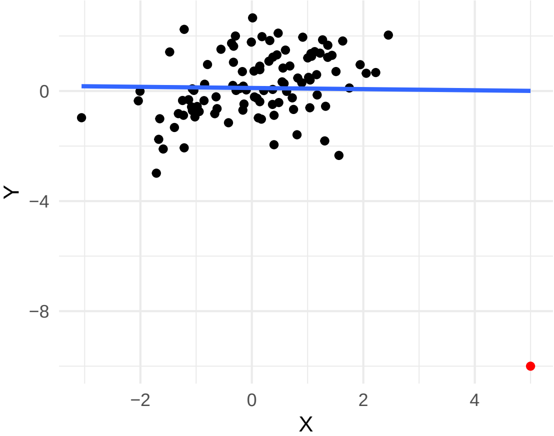
Model checking

Outliers: Outlier with low leverage



Model checking

Outliers: Outlier with high leverage



Model checking

Underfitting

- A model is **underfit** if it does not sufficiently explain the variance in the outcome.
- Consider the following population model:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + u$$

- The following model underfits because it does not account for the quadratic relationship:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x + u$$

- In short, we fail to observe the *signal* in the data (Molina and Garip 2019)

Model checking

Overfitting

- A model is **overfit** if it also explains *noise* in addition to the signal in the data (Molina and Garip 2019).
- Using the previous population, consider we estimate the following model:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x + \hat{\beta}_2 x^2 + \hat{\beta}_3 z + \hat{\beta}_4 z^2 + u$$

- The additional parameters $\hat{\beta}_3$ and $\hat{\beta}_4$ do not correspond to any of the population parameters, but may still explain some of the random variance in the outcome.

Model checking

Underfitting, overfitting, and generalization error

- Models suffering from under and overfitting will fail to generalize
  - Underfit models do not sufficiently explain variation in the outcome in the sample or population
  - Overfit models explain patterns in the sample that do not generalize to the population
- **Generalization error** refers to the expected prediction error when a model is applied to new data.

Model checking

Cross-validation

- **Cross-validation** is an approach used in machine-learning to assess the extent to which a predictive model can generalize to unseen data.
- The technique allows us to measure generalization error:
  1. Estimate an model using a sample X.
  2. Use the fitted model to predict the outcome for a new dataset X'.
  3. Compare the predictive accuracy (e.g. mean squared error) across the two datasets:
     - Model generalizes well if $MSE(\hat{y} = f(X)) \approx MSE(\hat{y} = f(X'))$ and both are low
     - Model overfits if $MSE(\hat{y} = f(X)) << MSE(\hat{y} = f(X'))$
     - Model underfit if both MSE scores are high

Model checking

Cross-validation

- Different kinds of cross-validation procedures are often used to evaluate generalization error:
  - **k-fold cross-validation**: data are split into k subsets. Models are estimated using $k-1$ subsets and predictions made for held-out set. Prediction error is averaged over k held-out sets.
  - **Leave-one-out cross-validation**: same procedure where each subset is a single datapoint. Requires estimation of N models.
  - **Temporal cross-validation**: Useful with panel data. Estimate a model with data from time t the assess predictions for data recorded at $t+1$.

Model checking

Cross-validation: loo

- The rstanarm package includes a function loo, which computes an approximation of LOO-CV, avoiding the need to fit N models.
- Models can be compared using the *expected log pointwise predictive density* (ELPD), a quantity that captures the predictive accuracy of the model (McElreath 7.2-4, GHV 11.8).
- The function can also be used to compute a k-fold CV score and an information theoretic measure WAIC.[3]

[3] See the documentation and vignette for further details on implementation.

Model checking

Cross-validation: LOO-CV

```
x <- rnorm(N)
z <- rnorm(N)
y <- 0.1*x + 2*(x^2) + rnorm(N)
df <- as.data.frame(y,x,z)

m <- stan_glm(y ~ x + I(x^2),  data = df,
                family = "gaussian", chains =1 , refresh = 0)
m.u <- stan_glm(y ~ x,  data = df,
                family = "gaussian", chains =1 , refresh = 0)
m.o <- stan_glm(y ~ x + I(x^2) + z + I(z^2),  data = df,
                family = "gaussian", chains =1 , refresh = 0)
```

Model checking

Cross-validation: LOO-CV

The loo function provides an approximation of the LOO-CV for an estimated model

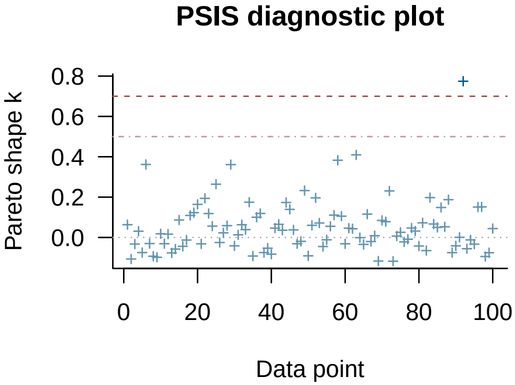
```
print(loo(m))
```

```
##
## Computed from 1000 by 100 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -145.8  7.0
## p_loo         3.5  0.7
## looic       291.5 14.0
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     98    98.0%   612
##  (0.5, 0.7]   (ok)        2     2.0%   460
```

Model checking

Cross-validation: LOO-CV

Individual points are scored using Pareto-Smoothed Importance Sampling (PSIS). High *pareto k* values ($k > .7$) indicate observations with high leverage.



Model checking

Cross-validation: LOO-CV

- `loo_compare` can be used to compare different models. The results rank the models from best to worst.

```
loo_compare(loo(m), loo(m.u), loo(m.o))
```

```
##      elpd_diff se_diff
## m       0.0       0.0
## m.o    -2.3       0.9
## m.u -118.4      14.9
```

Model checking

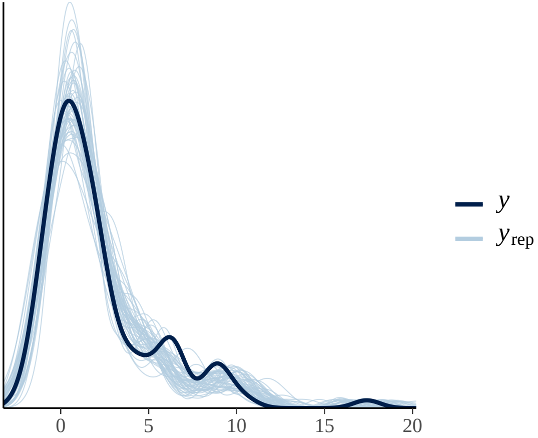
Posterior predictive checks

- We can also evaluate Bayesian models by examining the posterior predictive distribution
- Recall that Bayesian models are *generative*;
  - We can use the posterior distribution to make predictions, creating new, hypothetical datasets
- These predictions can be used to evaluate how well the models fit the data, including key statistics

Model checking

Posterior predictive checks

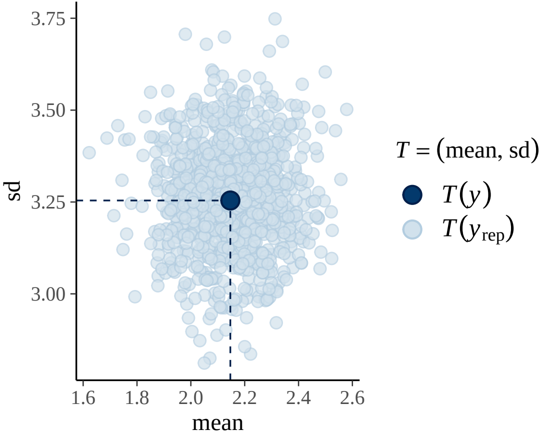
```
pp_check(m)
```



Model checking

Posterior predictive checks

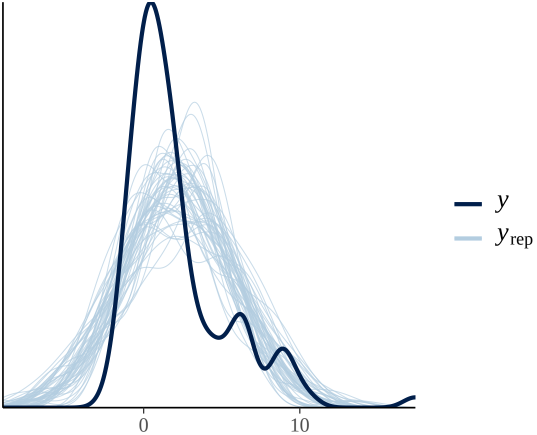
```
pp_check(m, plotfun = "stat_2d", stat = c("mean", "sd"))
```



Model checking

Posterior predictive checks

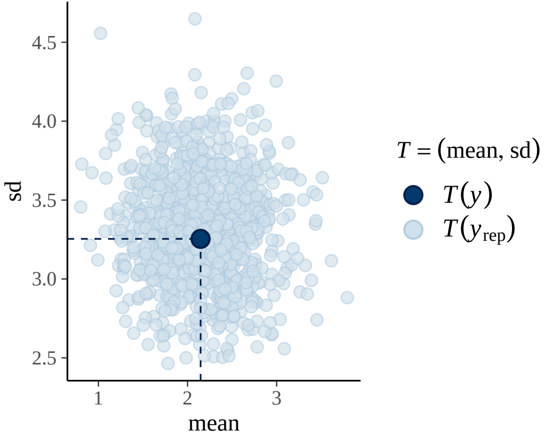
```
pp_check(m.u)
```



Model checking

Posterior predictive checks

```
pp_check(m.u, plotfun = "stat_2d", stat = c("mean", "sd"))
```



Model checking

Prediction and explanation

- Predictive models are designed to predict $\hat{y}$'s with high accuracy.
- In sociology, we typically estimate explanatory models where the primary goal is to estimate one or more $\hat{\beta}$'s and prediction is typically used to explore relationships between variables.
- Nonetheless, generalization error can be a useful heuristic for comparing and selecting different models, particularly when we do not have strong theory to guide model specification.[4]

[4] See Watts, Duncan J. 2014. "Common Sense and Sociological Explanations." American Journal of Sociology 120 (2): 313–51. https://doi.org/10.1086/678271 for further elaboration of this idea and Mullainathan, Sendhil, and Jann Spiess. 2017. "Machine Learning: An Applied Econometric Approach." Journal of Economic Perspectives 31 (2): 87–106. https://doi.org/10.1257/jep.31.2.87 for some discussion of the limitations of predictive modeling.

Model checking

Multicollinearity

- Recall that multicollinearity occurs when predictors are highly correlated and results in increased variance
- High pairwise correlations might indicate *potential* collinearity, but the issue can only be diagnosed after controlling for all relevant predictors

Model checking

Multicollinearity: VIF

- The **Variance Inflation Factor (VIF)** can be used to diagnose highly collinear predictors
- Consider the regression model $y = \beta_0 + \beta_i x_i + \beta_j x_j + ... + \beta_k x_k + u$
- A score is calculated for each *independent variable* using the following approach:
  - For x_i in $x_{i=1}, ..., x_k$, regress $x_i = \alpha_0 + \alpha_1 x_j + ... + \alpha_2 x_k + u$
  - Use R^2 from the model to calculate $VIF(\hat{\beta}_i) = \frac{1}{1-R_i^2}$
- VIF scores greater than $\approx 5 - 10$ indicate that a predict is highly collinear with one or more of the other predictors

Model checking

Multicollinearity: VIF

```
N <- 1000
x <- rnorm(N)
x2 <- rnorm(N)
x3 <- 0.8*x + rnorm(N)
y <- x + 2*x2 + 0.5*x3 + rnorm(N)

m <- lm(y ~ x + x2 + x3)
library(car)
vif(m)
```

```
##        x       x2       x3
## 1.660806 1.000244 1.660748
```

Model robustness

Defining robustness

- **Model robustness** refers to how *robust* the results of a given model are to alternative specifications.
  - The concern is that a result (such as $p < 0.05$) is sensitive to a particular specification of a model
- Often we try to mitigate such concerns by estimating several different specifications of a model

Model robustness

How robust are our results?

- Recent work calls for greater attention to specification issues as a way to address robustness concerns (Young and Holsteen 2017, Muñoz and Young 2018)
  - Critique: Reporting a handful of ad hoc specifications is insufficient to ensure robustness
  - Solution: Estimate models with *every possible combination of independent variables* and assess the distribution of coefficients
  - The goal is to explore the entire *model space* and to construct a distribution of estimates

Model robustness

How robust are our results?

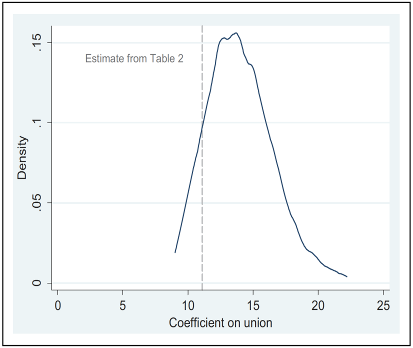


Figure 1. Modeling distribution of union wage premium.
Note: Kernel density graph of estimates from 1,024 models. Vertical line indicates the preferred estimate of an 11 percent union wage premium as reported in Table 2.

Young and Holsteen 2017.

Model robustness

How robust are our results?

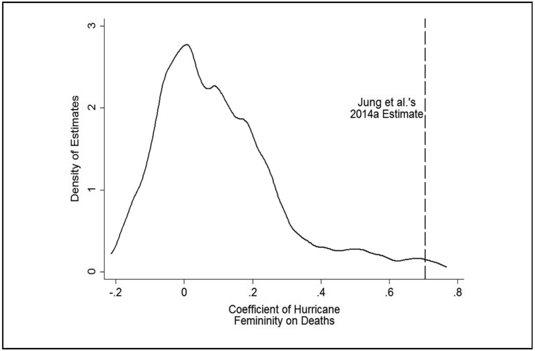


Figure 3. Model robustness results on Jung et al. (2014a) data.
Note: Kernel density graph of estimates from 1,152 models. See Table 6 for more information about the modeling distribution.

Muñoz and Young 2018.

Model robustness

Bayesian Model Averaging

- Bayesians have proposed a similar idea known as Bayesian Model Averaging
  - Estimate several models and construct an average across the models, weighted by the model fit, i.e. higher weights to better models
- There has been some debate about whether this approach is preferable to the Young-Holsteen-Muñoz technique.
  - The latter argue that it is problematic to weight different models if we do not know which is better a priori and that weighting requires more assumptions.[5]

[5] See Slez' 2017 comment on Young and Holsteen and the rejoinder by the latter and Bruce Western's 2018 comment on Muñoz and Young

Conclusions

- Missing data
  - Carefully examine any patterns of missing data
  - Choose an appropriate strategy to address the problem
- Model checking
  - Use diagnostic checks to identify and address potential issues with models
- Model robustness
  - Estimate multiple specifications to ensure results are robust

Next week

- Spring break!
- After spring break
  - Generalized linear models