Replication Code for How Governance and Disasters Shape Renewable Energy

Timothy Fraser, Northeastern University

Updated March 2, 2020

- 1. Import Data and Load Packages
- 3. Dispersion Tests
- 4. Modeling
    - 4.1 Total Solar Adoption
    - 4.2 Small Solar Farm Adoption
    - 4.3 Medium Solar Farm Adoption
    - 4.4 Large Solar Farm Adoption
    - 4.5 Extra-Large Solar Farm Adoption
- 5. Model Table
    - 5.1 Main Predictors Model Table
    - 5.2 Prefectures Model Table
    - 5.3 Model Statistics Table
- 6. Visualizations

This document replicates and revises the R script for the studied, “How Governance and Disasters shape Renewable Energy Transitions: The case of Japanese mega-solar,” which was conducted in August of 2018 and published in 2019 in Social Science Quarterly.

Recently, it came to my attention that I made an error in my coding for multiple imputation. As a result, I have updated the analysis, using mean imputation instead. Mean imputation is a fairly conservative way of replacing missing data, because it biases *against* small trends in your data. As a result, any statistically significant effects that we find tend to be strong trends in the data. This analysis confirms that the effects discussed in the paper are still consistent with the corrected, mean-imputed models.

Below, I exhibit my code and describe and explain its outputs, step by step.

1. Import Data and Load Packages

```
# Load packages
library(tidyverse) # for data visualization and wrangling
library(car) # for multicolinearity measures
library(BaylorEdPsych) # for GLM goodness of fit measures
library(gtools) # for getting stars from pvalues easily

# also please install pcsl and mfx, but don't load them
# (they require MASS, which conflicts with tidyverse's select function)

# Import dataset
dat <- readxl::read_excel("Database_prefectural_analysis_R_relevant_variables.xlsx") %>%
  select(
    # Import basic identifying information
    muni_code = Code,
    muni = Municipality_Japanese,
    pref = Prefecture_English,
    utility = Regional_Power_Company,

    # Import number of solar farms of various sizes,
    # and rename them to easily understandable names
    sp_total = SP,
    sp_small = SP_10_49_kW,
    sp_medium = SP_50_499_kW,
    sp_large = SP_500_1999kW,
    sp_extralarge = `SP_2000_kW+`,

    # Import predictors
    pv_output = PV_output_2018,
    area = Area_inhabitable_2010,
    pop = Population_2010,
    employment_energy = PercentEmployment_EGHW_2009,
    land_prices = Land_price_residential_2009,
    crime_rates = Crime_Rate_2008,
    voter_turnout = voter_turnout_2012,
    winning_party_voteshare = ku_LDP_voteshare_2012,
    unemployment = Unemployment_2010,
    income_per_capita = Income_taxable_per_capita_2010,
    ratio_revs_exp = Ratio_Revs_Exp_2010,
    carbon_emissions_change = PercentChange_CO2_industry_2000_03,
    deaths = death11,
    damages = damaged11,
    exclusion_zone = Fukushima_Exclusion_Zone) %>%
  # Convert categorical variables to factors, which will preserve their levels in models
  mutate(pref = as.factor(pref),
         exclusion_zone = factor(exclusion_zone, levels = c("1", "0")) )
```

```
# Calculate missing data (no variables over 5%)
dat %>%
  select(-c(muni_code, muni,pref,utility,exclusion_zone)) %>%
  summarise_all(funs(sum(is.na(.)) / n() * 100 )) %>%
  t() %>% as.data.frame() %>% rename(`(%) Missing` = V1)
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

	(%) Missing <dbl>
sp_total	0.0000000
sp_small	0.0000000
sp_medium	0.0000000
sp_large	0.0000000
sp_extralarge	0.0000000
pv_output	0.0000000
area	0.1148765
pop	0.0000000
employment_energy	0.0000000
land_prices	0.0000000

```
# Remove outcome values that are 0, so that we can model just the positive values

# Let's write a quick function so that when we scale our predictors,
# they come out as simple vectors, not matrices.
scale_fixed = function(x){
  scale(x) %>%
    as.data.frame() %>%
    unlist() %>%
    return()}

dat.count = dat %>%
  mutate_at(
    # exclude categories and outcome variables;
    # (note: no outcomes are missing data)
    vars(-c(muni_code, muni, pref, utility, exclusion_zone,
            sp_total, sp_small, sp_medium, sp_large, sp_extralarge)),
    # Replace missing data points in all predictors with the mean of the predictor
    funs(if_else(is.na(.), true = mean(., na.rm = TRUE), false = . ))) %>%
  # Now scale all predictors, so they are mean centered at 0
  # Doing so will allow us to compare the effects of numeric predictors
  mutate_at(
    vars(-c(muni_code, muni, pref, utility, exclusion_zone,
            sp_total, sp_small, sp_medium, sp_large, sp_extralarge)),
    funs(scale_fixed)) %>%
  # In order to use negative binomial models, let's left censor the data
  # so that all our zeros are replaced with NAs.
  mutate(
    sp_total = if_else(sp_total == 0, NA_real_, sp_total),
    sp_small = if_else(sp_small == 0, NA_real_, sp_small),
    sp_medium = if_else(sp_medium == 0, NA_real_, sp_medium),
    sp_large = if_else(sp_large == 0, NA_real_, sp_large),
    sp_extralarge = if_else(sp_extralarge == 0, NA_real_, sp_extralarge))
```

3. Dispersion Tests

```
# Is our data overdispersed?
dat %>%
  summarize(
    disp_total = var(sp_total) / mean(sp_total),
    disp_small = var(sp_small) / mean(sp_small),
    disp_medium = var(sp_medium) / mean(sp_medium),
    disp_large = var(sp_large) / mean(sp_large),
    disp_extralarge = var(sp_extralarge) / mean(sp_extralarge)) %>%
  pivot_longer(
    cols = -c(),
    names_to = "type",
    names_prefix = "disp_",
    values_to = "dispersion")
```

type <chr>	dispersion <dbl>
total	802.69506
small	643.45452
medium	65.97327
large	125.64353
extralarge	566.56014

5 rows

```
# Quite!
# Looks like the negative binomial model will probably be a better fit than poisson mode
ls.
```

4. Modeling

4.1 Total Solar Adoption

```
# Run a negative binomial model, using robust standard errors with marginal
nb.sp_total <- dat.count %>%
  mfx::negbinmfx(formula = sp_total ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Run a poisson model
p.sp_total <- dat.count %>%
  mfx::poissonmfx(formula = sp_total ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Does the Negative Binomial Model significantly improve upon the poisson?
data.frame(
  loglik_nb = logLik(nb.sp_total$fit),
  loglik_poisson = logLik(p.sp_total$fit)) %>%
  mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson), df = 1, lower.tail = FALSE))
```

loglik_nb <dbl>	loglik_poisson <dbl>	p.value <dbl>
-9922.093	-142544.5	0

1 row

Let's evaluate a Vuong test, which tests model fit using the AIC of both models.

```
# This test apparently uses a lot of computational power;
# had to suppress it in order to write to pdf.
# If you run it, you will find that all vuong test results are statistically significant
# below p < 0.001, and they all favor the negative binomial model.
# pscl::vuong(nb.sp_total$fit, p.sp_total$fit)
```

Looks like the negative binomial model has a statistically significant, better fit.

```
# Prefectures generate high VIFs for GVIF, but the appropriate way to compare categorica
l and numeric variables
# is instead to square the GVIF(1/2*DF), and compare those numbers.
# Very, very low VIF all around.
nb.sp_total$fit %>% vif()
```

```
##                              GVIF Df GVIF^(1/(2*Df))
## pv_output                3.301685  1        1.817054
## area                     1.905166  1        1.380277
## pop                      1.795431  1        1.339937
## employment_energy        1.171787  1        1.082491
## land_prices              2.887108  1        1.699149
## crime_rates              2.537045  1        1.592810
## winning_party_voteshare  2.133657  1        1.460704
## unemployment             1.664376  1        1.290107
## income_per_capita        3.698030  1        1.923026
## ratio_revs_exp           1.332405  1        1.154298
## carbon_emissions_change  1.080300  1        1.039375
## deaths                   1.282330  1        1.132400
## damages                  1.263960  1        1.124260
## exclusion_zone           1.301162  1        1.140685
## pref                   156.122079 46        1.056433
```

```
# Pretty good Nagelkerke's R2 - 0.77
nb.sp_total$fit %>% PseudoR2()
```

```
##         McFadden     Adj.McFadden        Cox.Snell       Nagelkerke
##       0.52045013       0.48872258       0.71091612       0.78305990
## McKelvey.Zavoina            Effron            Count        Adj.Count
##               NA     -10.57058122       0.04392923       0.01260239
##              AIC    Corrected.AIC
##    1996.21246904    2001.00891799
```

4.2 Small Solar Farm Adoption

```
# Run a negative binomial model, using robust standard errors with marginal
nb.sp_small <- dat.count %>%
  mfx::negbinmfx(formula = sp_small ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Run a poisson model
p.sp_small <- dat.count %>%
  mfx::poissonmfx(formula = sp_small ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Does the Negative Binomial Model significantly improve upon the poisson?
data.frame(
  loglik_nb = logLik(nb.sp_small$fit),
  loglik_poisson = logLik(p.sp_small$fit)) %>%
  mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson), df = 1, lower.tail = FALSE))
```

loglik_nb <dbl>	loglik_poisson <dbl>	p.value <dbl>
-9203.956	-96574.47	0

1 row

Let's evaluate a Vuong test, which tests model fit using the AIC of both models.

```
# This test apparently uses a lot of computational power;
# had to suppress it in order to write to pdf.
# Same as above.
# pscl::vuong(nb.sp_small$fit, p.sp_small$fit)
```

Looks like the negative binomial model has a statistically significant, better fit.

```
# Prefectures generate high VIFs for GVIF, but the appropriate way to compare categorica
l and numeric variables
# is instead to square the GVIF(1/2*DF), and compare those numbers.
# Very, very low VIF all around.
nb.sp_small$fit %>% vif()
```

```
##                              GVIF Df GVIF^(1/(2*Df))
## pv_output                3.277772  1        1.810462
## area                     1.934150  1        1.390737
## pop                      1.800456  1        1.341811
## employment_energy        1.172555  1        1.082846
## land_prices              2.898349  1        1.702454
## crime_rates              2.566529  1        1.602039
## winning_party_voteshare  2.128563  1        1.458959
## unemployment             1.675479  1        1.294403
## income_per_capita        3.779738  1        1.944155
## ratio_revs_exp           1.333598  1        1.154815
## carbon_emissions_change  1.078625  1        1.038569
## deaths                   1.282628  1        1.132532
## damages                  1.265678  1        1.125024
## exclusion_zone           1.286575  1        1.134273
## pref                   157.002958 46        1.056498
```

```
# Pretty good Nagelkerke's R2!
nb.sp_small$fit %>% PseudoR2()
```

```
##         McFadden     Adj.McFadden        Cox.Snell       Nagelkerke
##       0.56947855       0.54031018       0.77541372       0.83613038
## McKelvey.Zavoina           Effron            Count        Adj.Count
##               NA      -6.13077659       0.04935225       0.01659221
##              AIC    Corrected.AIC
##    1952.22424215    1957.07607025
```

4.3 Medium Solar Farm Adoption

```
# Run a negative binomial model, using robust standard errors with marginal
nb.sp_medium <- dat.count %>%
  mfx::negbinmfx(formula = sp_medium ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Run a poisson model
p.sp_medium <- dat.count %>%
  mfx::poissonmfx(formula = sp_medium ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Does the Negative Binomial Model significantly improve upon the poisson?
data.frame(
  loglik_nb = logLik(nb.sp_medium$fit),
  loglik_poisson = logLik(p.sp_medium$fit)) %>%
  mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson), df = 1, lower.tail = FALSE))
```

loglik_nb <dbl>	loglik_poisson <dbl>	p.value <dbl>
-4394.947	-10236.99	0

1 row

Let's evaluate a Vuong test, which tests model fit using the AIC of both models.

```
# This test apparently uses a lot of computational power;
# had to suppress it in order to write to pdf.
# Same as above.
# pscl::vuong(nb.sp_medium$fit, p.sp_medium$fit)
```

Looks like the negative binomial model has a statistically significant, better fit.

```
# Prefectures generate high VIFs for GVIF, but the appropriate way to compare categorica
l and numeric variables
# is instead to square the GVIF(1/2*DF), and compare those numbers.
# Very, very low VIF all around.
nb.sp_medium$fit %>% vif()
```

```
##                              GVIF Df GVIF^(1/(2*Df))
## pv_output                3.298480  1        1.816172
## area                     2.132005  1        1.460139
## pop                      1.936325  1        1.391519
## employment_energy        1.096720  1        1.047244
## land_prices              3.131260  1        1.769537
## crime_rates              2.376313  1        1.541529
## winning_party_voteshare  2.132058  1        1.460157
## unemployment             1.896027  1        1.376963
## income_per_capita        4.321734  1        2.078878
## ratio_revs_exp           1.443811  1        1.201587
## carbon_emissions_change  1.102750  1        1.050119
## deaths                   1.319646  1        1.148759
## damages                  1.311021  1        1.144998
## exclusion_zone           1.198751  1        1.094875
## pref                   176.649598 46        1.057853
```

```
# Pretty good Nagelkerke's R2!
nb.sp_medium$fit %>% PseudoR2()
```

```
##         McFadden     Adj.McFadden        Cox.Snell       Nagelkerke
##     5.100089e-01     4.623465e-01     6.687204e-01     7.552833e-01
## McKelvey.Zavoina           Effron            Count        Adj.Count
##               NA     1.254273e-01     1.815154e-01     4.932302e-02
##              AIC    Corrected.AIC
##     1.396777e+03     1.403418e+03
```

4.4 Large Solar Farm Adoption

```
# Run a negative binomial model, using robust standard errors with marginal
nb.sp_large <- dat.count %>%
  mfx::negbinmfx(formula = sp_large ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Run a poisson model
p.sp_large <- dat.count %>%
  mfx::poissonmfx(formula = sp_large ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Does the Negative Binomial Model significantly improve upon the poisson?
data.frame(
  loglik_nb = logLik(nb.sp_large$fit),
  loglik_poisson = logLik(p.sp_large$fit)) %>%
  mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson), df = 1, lower.tail = FALSE))
```

loglik_nb <dbl>	loglik_poisson <dbl>	p.value <dbl>
-5663.776	-26835.99	0

1 row

Let's evaluate a Vuong test, which tests model fit using the AIC of both models.

```
# This test apparently uses a lot of computational power;
# had to suppress it in order to write to pdf.
# Same as above.
# pscl::vuong(nb.sp_large$fit, p.sp_large$fit)
```

Looks like the negative binomial model has a statistically significant, better fit.

```
# Prefectures generate high VIFs for GVIF, but the appropriate way to compare categorica
l and numeric variables
# is instead to square the GVIF(1/2*DF), and compare those numbers.
# Very, very low VIF all around.
nb.sp_large$fit %>% vif()
```

```
##                              GVIF Df GVIF^(1/(2*Df))
## pv_output                3.426410  1        1.851057
## area                     1.967373  1        1.402631
## pop                      1.941090  1        1.393230
## employment_energy        1.218389  1        1.103807
## land_prices              3.092172  1        1.758457
## crime_rates              2.449533  1        1.565099
## winning_party_voteshare  2.125701  1        1.457978
## unemployment             1.722320  1        1.312372
## income_per_capita        4.220310  1        2.054339
## ratio_revs_exp           1.430219  1        1.195918
## carbon_emissions_change  1.142522  1        1.068888
## deaths                   1.280909  1        1.131773
## damages                  1.286797  1        1.134371
## exclusion_zone           1.415970  1        1.189945
## pref                   153.018708 46        1.056203
```

```
# Pretty good Nagelkerke's R2!
nb.sp_large$fit %>% PseudoR2()
```

```
##         McFadden     Adj.McFadden        Cox.Snell       Nagelkerke
##       0.39372843       0.34074736       0.51818465       0.61434228
## McKelvey.Zavoina           Effron            Count        Adj.Count
##               NA      -1.17500039       0.14104596       0.04409171
##              AIC    Corrected.AIC
##    1540.95376368    1547.25709701
```

4.5 Extra-Large Solar Farm Adoption

```
# Run a negative binomial model, using robust standard errors with marginal
nb.sp_extralarge <- dat.count %>%
  mfx::negbinmfx(formula = sp_extralarge ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Run a poisson model
p.sp_extralarge <- dat.count %>%
  mfx::poissonmfx(formula = sp_extralarge ~ pv_output + area + pop +
                         employment_energy + land_prices +
                         crime_rates + winning_party_voteshare +
                         unemployment + income_per_capita +
                         ratio_revs_exp + carbon_emissions_change +
                         deaths + damages + exclusion_zone + pref, robust = TRUE)

# Does the Negative Binomial Model significantly improve upon the poisson?
data.frame(
  loglik_nb = logLik(nb.sp_extralarge$fit),
  loglik_poisson = logLik(p.sp_extralarge$fit)) %>%
  mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson), df = 1, lower.tail = FALSE))
```

loglik_nb <dbl>	loglik_poisson <dbl>	p.value <dbl>
-2298.894	-36223.14	0

1 row

Let's evaluate a Vuong test, which tests model fit using the AIC of both models.

```
# This test apparently uses a lot of computational power;
# had to suppress it in order to write to pdf.
# Same as above.
#pscl::vuong(nb.sp_extralarge$fit, p.sp_extralarge$fit)
```

Looks like the negative binomial model has a statistically significant, better fit.

```
# Prefectures generate high VIFs for GVIF, but the appropriate way to compare categorica
l and numeric variables
# is instead to square the GVIF(1/2*DF), and compare those numbers.
# Very, very low VIF all around.
nb.sp_extralarge$fit %>% vif()
```

```
##                               GVIF Df GVIF^(1/(2*Df))
## pv_output                 3.367556  1        1.835090
## area                      2.159413  1        1.469494
## pop                       2.556221  1        1.598819
## employment_energy         1.446296  1        1.202620
## land_prices               3.471109  1        1.863091
## crime_rates               3.056317  1        1.748232
## winning_party_voteshare   1.844700  1        1.358197
## unemployment              1.936716  1        1.391660
## income_per_capita         3.635615  1        1.906729
## ratio_revs_exp            1.476419  1        1.215080
## carbon_emissions_change   1.135331  1        1.065519
## deaths                    1.356094  1        1.164514
## damages                   1.354547  1        1.163850
## exclusion_zone            1.656029  1        1.286868
## pref                    162.840302 45        1.058218
```

```
# Pretty good Nagelkerke's R2!
nb.sp_extralarge$fit %>% PseudoR2()
```

```
##         McFadden     Adj.McFadden        Cox.Snell       Nagelkerke
##       0.30414954       0.16056160       0.40985753       0.49774968
## McKelvey.Zavoina           Effron            Count        Adj.Count
##               NA       0.13974053       0.05102041      -0.30617978
##              AIC    Corrected.AIC
##     711.23180610     728.29474316
```

```
# Save models
save(nb.sp_total, nb.sp_small, nb.sp_medium, nb.sp_large, nb.sp_extralarge,
     p.sp_total, p.sp_small, p.sp_medium, p.sp_large, p.sp_extralarge,
     file = "models.RData")
# Remove models, getting rid of poisson models for good.
remove(nb.sp_total, nb.sp_small, nb.sp_medium, nb.sp_large, nb.sp_extralarge,
       p.sp_total, p.sp_small, p.sp_medium, p.sp_large, p.sp_extralarge)
```

5. Model Table

5.1 Main Predictors Model Table

<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN” “http://www.w3.org/TR/html4/loose.dtd (http://www.w3.org/TR/html4/loose.dtd)”>

	Total Solar	**Small (10-49 kW)**	**Medium (50-499 kW)**	**Large (500-1999 kW)**	**Extra-Large (+2000 kW)**
Photovoltaic Output (2018)	**63.26*****	**40.33***	**4.45*****	**6.04****	3.49
	(16.63)	(15.91)	(0.88)	(1.89)	(6.25)
Inhabitable area (2010)	**119.48*****	**81.83*****	**7.95*****	**20.60*****	**7.84***
	(6.11)	(4.68)	(0.48)	(1.44)	(3.24)

Population (2010)	**16.31****	**9.29***	0.50	0.04	**-9.32*****
	(5.67)	(4.31)	(0.35)	(1.15)	(2.78)
% Employment in electricity, gas, water, or heating (2009)	-1.55	**-11.35***	**-1.07***	-3.21	**-9.48****
	(13.35)	(4.78)	(0.53)	(1.65)	(3.43)
Crime Rate (2008)	2.27	6.45	**1.62***	**4.28***	5.92
	(7.30)	(4.18)	(0.63)	(1.84)	(9.80)
% Liberal Democratic Party (LDP) vote share in 2012 Upper House election	-1.56	-4.84	-0.50	1.32	0.04
	(5.74)	(4.10)	(0.56)	(1.50)	(4.18)
Unemployment Rate (2010)	**58.18*****	**39.46*****	**1.35***	**7.51*****	9.20
	(9.05)	(6.60)	(0.54)	(1.37)	(6.26)
Taxable Income per capita (2010)	-12.44	-9.28	1.14	4.09	**30.53***
	(7.75)	(5.07)	(1.00)	(2.28)	(12.25)
Local government revenues to expenditures ratio (2010)	**-18.82****	**-15.18****	**-1.44****	-0.47	0.56
	(6.82)	(5.22)	(0.50)	(1.39)	(4.16)
% Increase in Industrial CO2 emissions (2000-2003)	-6.17	**-8.86***	0.39	2.63	-1.38
	(6.54)	(4.21)	(0.43)	(2.64)	(3.44)
Deaths in 2011 disaster	5.15	3.24	0.24	1.26	2.59
	(4.42)	(2.65)	(0.28)	(1.04)	(2.04)
Damaged properties in 2011 disaster	**-6.79****	**-3.23***	-0.01	-0.19	1.09
	(2.37)	(1.37)	(0.14)	(0.44)	(1.27)
Fukushima Exclusion Zone	-138.39	-56.10	-9.67	-40.75	-96.35
	(107.55)	(59.39)	(15.66)	(41.16)	(62.60)
Num. obs.	1639	1621	1201	1262	490
Log Likelihood	-9922.09	-9203.96	-4394.95	-5663.78	-2298.89
Deviance	1874.21	1830.22	1274.78	1418.95	591.23
AIC	19968.19	18531.91	8913.89	11451.55	4719.79
BIC	20303.10	18866.14	9229.53	11770.26	4975.65

*** denotes statistically significant results at p < 0.001, ** at p < 0.01, and * at 0 < 0.05. Estimates are marginal effects, read as the increase in actual solar farms given an increase of one standard deviation in the predictor.

Statistical models

5.2 Prefectures Model Table

<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN” “http://www.w3.org/TR/html4/loose.dtd (http://www.w3.org/TR/html4/loose.dtd)”>

	Total Solar	**Small (10-49 kW)**	**Medium (50-499 kW)**	**Large (500-1999 kW)**	**Extra-Large (+2000 kW)**
Hokkaido	**-141.01*****	**-99.80*****	**-11.32*****	**-15.50***	92.43
	(10.72)	(7.38)	(1.01)	(6.40)	(67.05)
Akita	**-113.83*****	**-84.33*****	**-7.96****	0.15	199.42
	(27.92)	(20.89)	(2.68)	(12.20)	(201.53)
Aomori	**-119.78*****	**-86.30*****	**-11.57*****	3.84	751.06
	(22.84)	(17.54)	(1.14)	(12.41)	(537.56)

Iwate	**-128.71*****	**-95.92*****	**-11.65*****	-11.10	552.30
	(17.13)	(11.65)	(0.94)	(8.13)	(325.79)
Miyagi	-60.22	**-83.29*****	**-10.37*****	23.50	1600.87
	(43.28)	(14.13)	(1.07)	(17.53)	(1432.26)
Fukushima	-54.64	**-81.70*****	**-7.41*****	30.83	1321.84
	(42.31)	(17.18)	(1.88)	(21.39)	(759.58)
Yamagata	-63.97	**-89.55*****	**-6.51***	19.08	3713.51
	(46.38)	(15.16)	(2.83)	(21.10)	(2306.75)
Ibaraki	**124.87****	**54.82***	2.63	**74.87****	**642.17***
	(40.20)	(26.13)	(2.66)	(26.39)	(326.17)
Tochigi	75.33	37.37	0.23	26.52	763.46
	(38.61)	(25.93)	(3.69)	(14.25)	(414.97)
Gunma	49.17	40.02	5.18	20.65	1064.70
	(41.56)	(30.26)	(3.73)	(14.62)	(618.71)
Saitama	**-43.98***	**-46.76*****	-2.07	32.11	1761.60
	(22.20)	(13.31)	(2.35)	(19.13)	(1112.30)
Chiba	**84.38***	39.53	0.37	**32.12***	882.69
	(37.55)	(25.37)	(2.57)	(14.84)	(650.34)
Kanagawa	**-94.00*****	**-69.93*****	**-8.97*****	8.77	1033.18
	(18.28)	(13.30)	(1.23)	(13.64)	(960.30)
Tokyo	**-152.85*****	**-109.37*****	**-13.13*****	**-27.38*****	**-44.05*****
	(10.91)	(7.30)	(0.59)	(3.39)	(4.33)
Niigata	**-120.15*****	**-101.18*****	**-12.97*****	-13.10	5524.65
	(25.76)	(10.55)	(0.62)	(8.50)	(4785.91)
Nagano	**-95.93*****	**-69.37*****	**-4.14***	-1.01	538.23
	(13.16)	(9.06)	(1.92)	(8.39)	(343.32)
Yamanashi	**-48.77***	**-31.67***	**-9.41*****	12.81	46.09
	(21.99)	(15.53)	(1.19)	(15.31)	(63.35)
Shizuoka	52.03	5.45	7.82	**67.22***	981.87
	(34.39)	(20.76)	(4.15)	(28.67)	(526.71)
Toyama	**-96.15*****	**-73.38*****	-1.65	-16.60	**-36.93*****
	(25.41)	(17.70)	(3.72)	(9.56)	(7.63)
Ishikawa	-21.37	-40.80	-0.35	14.37	830.87
	(53.96)	(32.30)	(4.65)	(15.66)	(692.78)
Fukui	8.43	-46.86	3.24	24.74	
	(77.85)	(30.53)	(5.47)	(19.09)	
Gifu	45.94	38.23	-0.59	2.89	83.24
	(30.14)	(23.20)	(2.50)	(9.63)	(81.73)
Shiga	53.36	-3.96	**16.46***	**55.14***	919.85
	(56.13)	(33.68)	(7.32)	(26.26)	(902.79)
Mie	**117.60***	63.95	2.86	**65.77***	489.34
	(48.21)	(33.26)	(3.78)	(27.15)	(279.55)
Kyoto	-6.05	-44.24	1.01	75.11	433.21
	(56.61)	(30.86)	(4.35)	(40.88)	(439.99)
Nara	-57.06	**-48.50***	3.45	17.90	-5.69
	(33.29)	(24.55)	(3.78)	(15.94)	(23.97)
Wakayama	**-57.41***	**-47.28****	0.19	-3.76	537.07
	(23.91)	(17.22)	(3.37)	(7.87)	(340.63)

Osaka	**-131.79*****	**-97.63*****	**-8.99*****	**-13.36***	**-34.47****
	(12.94)	(9.54)	(1.51)	(6.71)	(10.75)
Hyogo	27.94	-12.13	**13.61****	**71.72****	230.63
	(42.57)	(27.90)	(4.68)	(25.55)	(136.24)
Tokushima	-16.45	-21.08	22.35	50.83	175.17
	(41.19)	(28.00)	(12.31)	(29.72)	(162.22)
Kagawa	73.81	16.23	**28.78****	**97.58****	371.26
	(45.18)	(27.21)	(10.44)	(35.53)	(265.89)
Kochi	**-115.95*****	**-89.27*****	-0.43	37.79	235.17
	(15.97)	(9.97)	(3.30)	(28.20)	(269.12)
Ehime	-43.58	**-46.93***	6.06	**46.89***	79.64
	(32.33)	(21.65)	(4.98)	(22.80)	(85.48)
Okayama	72.85	-2.19	-0.93	**44.18***	2272.44
	(53.31)	(26.84)	(3.04)	(21.29)	(1287.49)
Tottori	-51.40	**-61.81***	-4.25	**71.15***	542.37
	(46.45)	(26.95)	(3.37)	(35.33)	(543.22)
Shimane	**-88.61****	**-82.09*****	-2.66	49.78	192.70
	(28.23)	(15.80)	(3.77)	(27.69)	(184.56)
Hiroshima	16.30	8.38	7.51	**42.71***	471.99
	(41.56)	(30.51)	(4.56)	(21.71)	(368.22)
Yamaguchi	14.88	**-46.02***	-3.29	**71.61***	2287.58
	(47.50)	(19.95)	(2.63)	(36.10)	(1313.75)
Fukuoka	-53.69	**-51.26***	**-5.32****	23.69	302.37
	(32.28)	(23.34)	(2.04)	(16.22)	(195.10)
Oita	81.09	55.20	**-7.78*****	38.41	1152.24
	(68.12)	(49.03)	(1.74)	(20.05)	(674.15)
Saga	-36.31	-24.13	-3.16	-2.09	309.17
	(35.49)	(27.60)	(2.79)	(10.51)	(256.84)
Nagasaki	18.36	6.39	-3.37	**44.92***	264.39
	(69.46)	(53.47)	(3.22)	(21.48)	(197.58)
Kumamoto	-46.55	**-44.30***	**-8.17*****	37.66	1399.72
	(27.94)	(18.98)	(1.41)	(21.41)	(825.34)
Miyazaki	-44.41	**-44.10***	**-6.42****	28.51	3725.23
	(31.50)	(20.20)	(2.43)	(17.90)	(2034.11)
Kagoshima	-9.66	-24.54	-1.07	**92.86***	1396.39
	(41.69)	(27.91)	(3.03)	(40.61)	(837.07)
Okinawa	**-116.61*****	**-77.73*****	**-6.27****	-7.58	**-35.36*****
	(20.37)	(17.51)	(2.42)	(13.02)	(10.68)
Num. obs.	1639	1621	1201	1262	490
Log Likelihood	-9922.09	-9203.96	-4394.95	-5663.78	-2298.89
Deviance	1874.21	1830.22	1274.78	1418.95	591.23
AIC	19968.19	18531.91	8913.89	11451.55	4719.79
BIC	20303.10	18866.14	9229.53	11770.26	4975.65

*** denotes statistically significant results at p < 0.001, ** at p < 0.01, and * at 0 < 0.05. Estimates are marginal effects, read as the increase in actual solar farms given an increase of one standard deviation in the predictor.

Statistical models

5.3 Model Statistics Table

```
load("models.RData")
# Let's write functions to gather several model statistics:


# Number of Cases
samplesize = function(model){
  nobs(model$fit) %>%
    as.data.frame() %>%
    select(sample = 1) %>%
    return()
}

# Mean VIF
gvif = function(model){
model$fit %>%
  # Get the variance inflation factors
  vif() %>% data.frame() %>%
  # Grab the generalized variance inflation factors, raised to the 1/(2 * df) power,
    # This measure is comparable among both categorical and numeric variables
  select(gvif1_2 = 3) %>%
  # This measure can then be squared to get a measure along the same scale of the origin
al VIF
  # as described by Fox & Monette
  summarize(`Mean VIF` = mean(gvif1_2^2)) %>%
    return()
}

# Pseudo R2
nagelkerke = function(model){
model$fit %>% PseudoR2() %>%
    as.data.frame() %>%
    slice(4) %>%
  select(`Nagelkerke's Pseudo R2` = 1) %>%
    return()
}


# AIC
aic = function(model){
  model$fit %>%
    AIC() %>%
    as.data.frame() %>%
    select(`AIC` = 1) %>%
    return()
}


gof = bind_cols(

  # Model Name
  data.frame(Model = c("Total Solar", "Small Solar",
                       "Medium Solar", "Large Solar",
                       "Extra-Large Solar")),
```

```
# Sample Size
list(nb.sp_total, nb.sp_small, nb.sp_medium, nb.sp_large, nb.sp_extralarge) %>%
  lapply(samplesize) %>% bind_rows(),
# GVIF
list(nb.sp_total, nb.sp_small, nb.sp_medium, nb.sp_large, nb.sp_extralarge) %>%
  lapply(gvif) %>% bind_rows(),
# Nagelkerke's R2
list(nb.sp_total, nb.sp_small, nb.sp_medium, nb.sp_large, nb.sp_extralarge) %>%
  lapply(nagelkerke) %>% bind_rows(),
# AIC
list(nb.sp_total, nb.sp_small, nb.sp_medium,
     nb.sp_large, nb.sp_extralarge) %>%
  lapply(aic) %>% bind_rows(),

# Chi Square Test P-value
# How much does Negative Binomial improve upon Poisson Model?
bind_rows(

  data.frame(
    loglik_nb = logLik(nb.sp_total$fit),
    loglik_poisson = logLik(p.sp_total$fit)) %>%
    mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson),
                            df = 1, lower.tail = FALSE)) %>%
    select(p.value),

  data.frame(
    loglik_nb = logLik(nb.sp_small$fit),
    loglik_poisson = logLik(p.sp_small$fit)) %>%
    mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson),
                            df = 1, lower.tail = FALSE)) %>%
    select(p.value),

  data.frame(
    loglik_nb = logLik(nb.sp_medium$fit),
    loglik_poisson = logLik(p.sp_medium$fit)) %>%
    mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson),
                            df = 1, lower.tail = FALSE)) %>%
    select(p.value),

  data.frame(
    loglik_nb = logLik(nb.sp_large$fit),
    loglik_poisson = logLik(p.sp_large$fit)) %>%
    mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson),
                            df = 1, lower.tail = FALSE)) %>%
    select(p.value),

  data.frame(
    loglik_nb = logLik(nb.sp_extralarge$fit),
    loglik_poisson = logLik(p.sp_extralarge$fit)) %>%
    mutate(p.value = pchisq(2 * (loglik_nb - loglik_poisson),
                            df = 1, lower.tail = FALSE)) %>%
    select(p.value)))
```

```
# View goodness of fit statistics
gof
```

Model <fctr>	sample <int>	Mean VIF <dbl>	Nagelkerke's Pseudo R2 <dbl>	AIC <dbl>	p.value <dbl>
Total Solar	1639	1.898033	0.7830599	19968.186	0
Small Solar	1621	1.906459	0.8361304	18531.913	0
Medium Solar	1201	1.987730	0.7552833	8913.893	0
Large Solar	1262	1.989019	0.6143423	11451.553	0
Extra-Large Solar	490	2.104813	0.4977497	4719.788	0

5 rows

```
# Remove excess data
remove(p.sp_extralarge, p.sp_large, p.sp_medium, p.sp_small, p.sp_total,
       gof, aic, gvif, nagelkerke)
```

6. Visualizations

Finally, let’s visualize the most statistically significant effects.

First, we’ll write a function to generate mean values for most predictors, off of which we can predict solar adoption.

```
# What if we could see how changes in our independent variables
# affect the predicted amount of solar we receive?

# To do so, let's create a fake dataset
# where all continuous variables are held at their mean,
# and the prefecture category is set to just one value,
# except for one key variable we wish to see change in

sim = function(variable){

# Find the min and max of the variable you want to assess
bounds = dat.count %>%
  select(!!sym(variable)) %>%
  summarize(min = min(!!sym(variable)),
            max = max(!!sym(variable)))


# Create a data.frame with a range of values within those bounds
data.frame(
  id = variable,
  # If you choose, you can use those bounds to define the prediction
  # variable = seq(bounds$min, bounds$max, length.out = 100)) %>%
  # Or, you can set them to a more meaningful deviation around the mean, like 2 standard
deviations
  variable = seq(-2, 2, length.out = 100)) %>%
  rename(!!sym(variable) := variable) %>%
  # Join in the mean of all variables in the model
  left_join(by = "id",
            y = dat.count %>%
              summarize_at(
                # Calculate the mean for all except these variables
                vars(-c(muni_code, muni, pref, utility, exclusion_zone,
                        sp_total, sp_small, sp_medium,
                        sp_large, sp_extralarge,
                        !!sym(variable))),
                funs(mean)) %>%
              # Be sure to add in a prefecture
              mutate(pref = factor("Saitama", levels = levels(dat.count$pref)),
                     exclusion_zone = factor("0", levels = c("1" ,"0"))) %>%
              # And add an ID - the name of the variable that was varied - to complete t
he join
              mutate(id = variable)) %>%
  return()
}
```

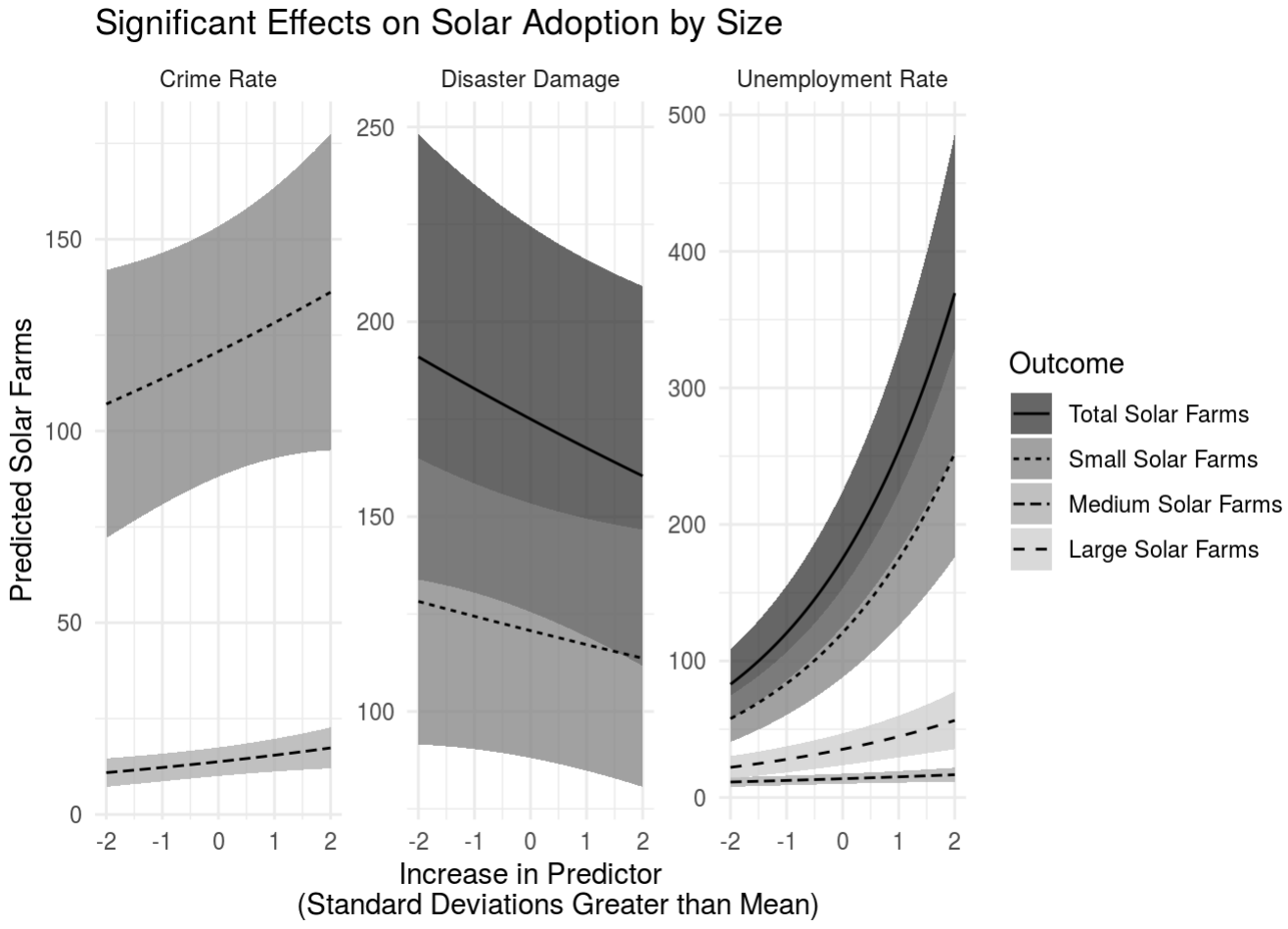
Generate data, varying by crime rate, damages, and unemployment, and predict the number of solar farms adopted.

```
# Now, let's simulate varying value for crime rates, etc.
simdata = c("crime_rates", "damages", "unemployment") %>%
  lapply(sim) %>% bind_rows()

# Write a function to predict solar farms
predict.model = function(model){
  predict(model$fit,
          newdata = simdata,
          type = "response",
          se.fit = TRUE) %>%
    as.data.frame() %>%
    # Bind in simulated data
    bind_cols(simdata) %>%
    # Add dependent variable of model
    mutate(model = names(model$fit$model)[1])
}


# Generate predictions
pred = list(nb.sp_total, nb.sp_small, nb.sp_medium,
            nb.sp_large, nb.sp_extralarge) %>%
  lapply(predict.model) %>%
  bind_rows() %>%
  # Pivot each predictor into a single column
  pivot_longer(
    cols = c(crime_rates, damages, unemployment),
    names_to = "predictor",
    values_to = "value") %>%
  # Then keep only the predictor values from each one's predictions
  # (we don't want to plot the mean values of that predictor)
  filter(predictor == id) %>%
  # Now identify each predicted value as significant or not
  # Sadly, we must do this manually,
  # specifying for which models which variables were significant.
  mutate(significant = if_else(
      # Damage was significant for total and small
      model %in% c("sp_total", "sp_small") & predictor == "damages", 1,
    if_else(
      # Unemployment was significant for all but extra-large
      !model %in% c("sp_extralarge") & predictor == "unemployment", 1,
      if_else(
         # Crime rates were significant for small and medium
        model %in% c("sp_small", "sp_medium") & predictor == "crime_rates", 1, 0)))) %>%
  filter(significant == 1) %>%
  # Relabel and arrange labels as we want to see them in the chart
  mutate(
    Outcome = model %>% recode_factor(
    "sp_total" = "Total Solar Farms",
    "sp_small" = "Small Solar Farms",
    "sp_medium" = "Medium Solar Farms",
    "sp_large" = "Large Solar Farms",
    "sp_extralarge" = "Extra-Large Solar Farms"),
    predictor = predictor %>% recode_factor(
```

```
      "crime_rates" = "Crime Rate",
      "damages" = "Disaster Damage",
      "unemployment" = "Unemployment Rate"))
```

Now visualize. The x-axis will have to be in terms of the scaled independent variables, such that an increase in 1 on the axis is an increase in that variable by 1 standard deviation from the mean.

```
# Visualize
pred %>%
  ggplot(mapping = aes(x = value, y = fit,
                       ymin = fit - se.fit*1.96,
                       ymax = fit + se.fit*1.96,
                       fill = Outcome,
                       linetype = Outcome)) +
  scale_fill_grey() +
  geom_ribbon(alpha = 0.75) +
  geom_line(color = "black") +
  theme_minimal() +
  facet_wrap(~predictor,scales = "free") +
  labs(x = "Increase in Predictor
       (Standard Deviations Greater than Mean)",
       y = "Predicted Solar Farms",
       title = "Significant Effects on Solar Adoption by Size")
```