Poisson Generalized Linear Model

Filippo Gambarota

University of Padova

2022/2023

Updated on 2023-05-07

Outline

Poisson GLM

Everything that we discussed for the binomial GLM is also relevant for the Poisson GLM. We are gonna focus on specificity of the Poisson model in particular:

- **Poisson distribution** and **link function**
- **Parameters interpretation**
- **Overdispersion** causes, consequences and remedies

Poisson distribution

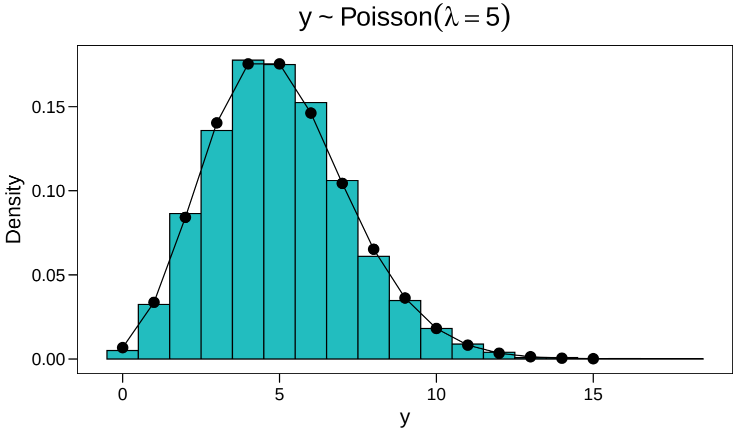
Poisson distribution

The Poisson distribution is defined as:

$$p(y) = \frac{e^{-\mu}\mu^{y}}{y!}$$

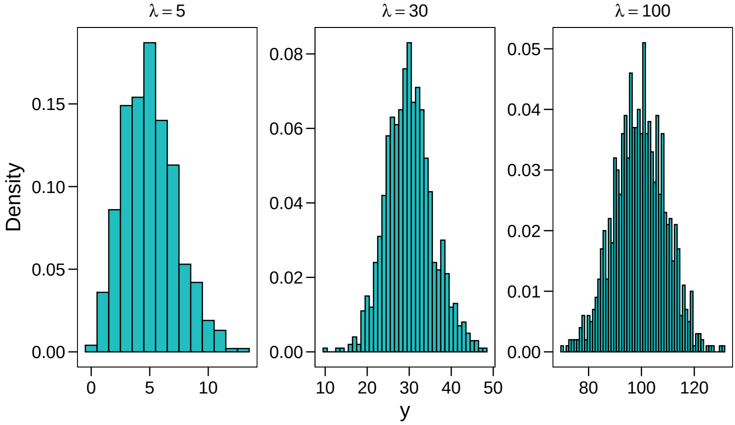
Where the mean is μ and the variance is μ

Poisson distribution

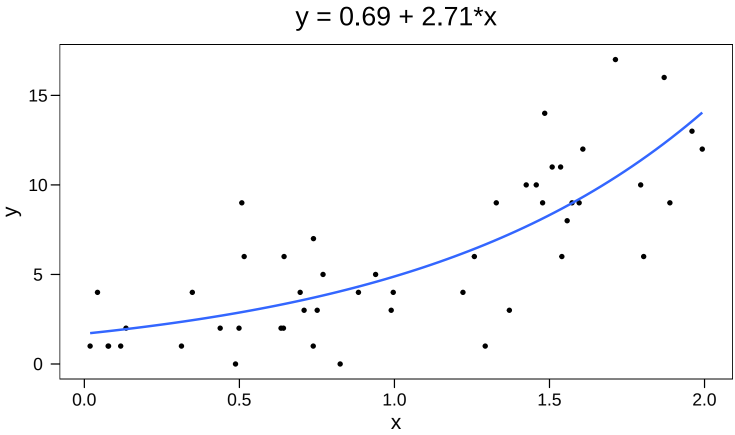


Poisson distribution

As the mean increases also the variance increase and the distributions is approximately normal:



Poisson distribution



Link function

The common (and default in R) link function ($g(\lambda)$) for the Poisson distribution is the **log** link function and the inverse of link function is the exponential.

$$log(\lambda) = \beta_0 + \beta_1 X_1 + ...\beta_p X_p$$
$$\lambda = e^{\beta_0 + \beta_1 X_1 + ...\beta_p X_p}$$

Parameters intepretation

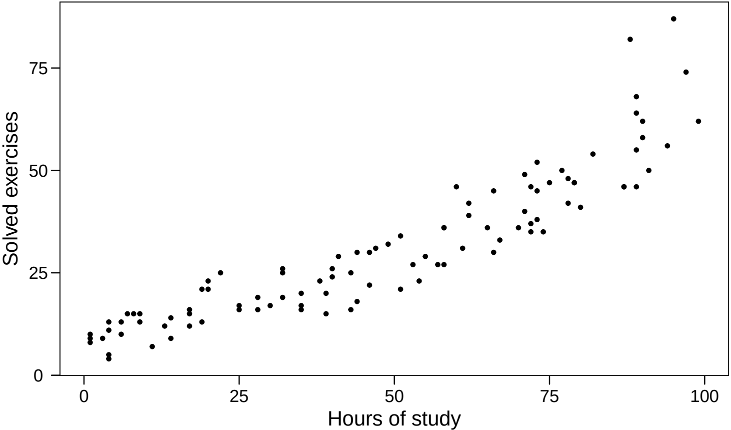
An example...

Let's start by a simple example trying to explain the number of errors of math exercises by students ($N = 100$) as a function of the number of hours of study.

id	**studyh**	**solved**
1	51	34
2	19	13
3	19	21
4	20	21
...	...	...
97	99	62
98	54	23
99	71	40
100	73	38

An example...

- There is a clear non-linear pattern
- There seems to be a positive relationship
- The variance seems to increase as a function of the predictor (remember that mean and variance are the same value for the Poisson)



Model fitting

We can fit the model using the `glm` function in R setting the appropriate **random component** (`family`) and the **link function** (`link`):

```r
fit <- glm(solved ~ studyh, family = poisson(link = "log"), data = dat)

summary(fit)
```

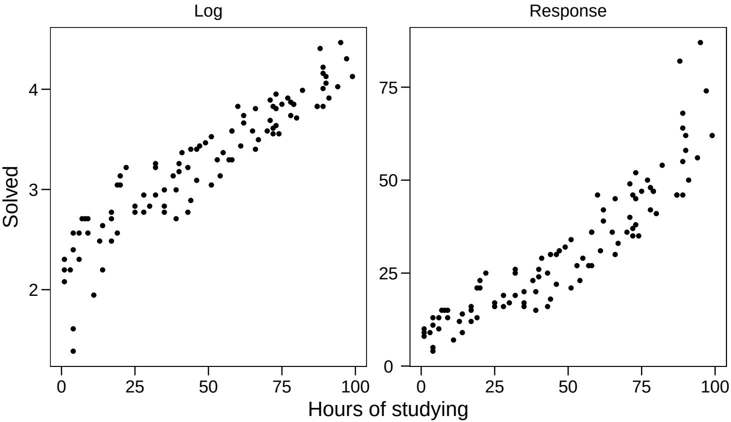
```
##
## Call:
## glm(formula = solved ~ studyh, family = poisson(link = "log"),
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4949  -0.8010  -0.1196   0.6968   3.0477
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 2.3425585  0.0472513   49.58   <2e-16 ***
## studyh      0.0194047  0.0006888   28.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 986.59  on 99  degrees of freedom
## Residual deviance: 110.76  on 98  degrees of freedom
## AIC: 624.64
##
## Number of Fisher Scoring iterations: 4
```

Parameters intepretation

- The `(Intercept)` 2.343 is the log of the expected number of solved exercises for a student with 0 hours of studying. Taking the exponential we obtain the estimation on the response scale 10.408
- the `studyh` 0.019 is the increase in the expected increase of (log) solved exercises for a unit increase in hours of studying. Taking the exponential we obtain the ratio of increase of the number of solved exercises 1.020

Parameters intepretation

Again, as in the binomial model, the effects are linear on the log scale but non-linear on the response scale.



Parameters intepretation

The non-linearity can be easily seen using the `predict()` function:

```
linear <- predict(fit, newdata = data.frame(studyh = c(10, 11)))
diff(linear) # same as the beta0
```

```
##          2
## 0.01940473
```

```
nonlinear <- exp(linear) # or predict(..., type = "response")
diff(nonlinear)
```

```
##         2
## 0.2476064
```

```
# ratio of increase when using the response scale
nonlinear[2]/nonlinear[1]
```

```
##        2
## 1.019594
```

```
# equivalent to exp(beta1)
exp(coef(fit)[2])
```

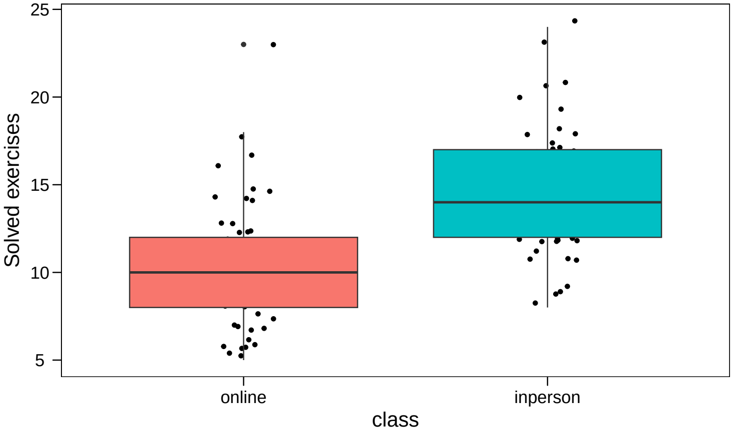
```
##   studyh
## 1.019594
```

Parameters intepretation - Categorical variable

Let's make a similar example with the number of solved exercises comparing students who attended online classes and students attending in person.

id	class	class.c	solved
1	online	0	9
2	inperson	1	12
3	online	0	10
4	inperson	1	8
...	...	...	...
97	online	0	9
98	inperson	1	18
99	online	0	10
100	inperson	1	15

Parameters intepretation - Categorical variable



Parameters intepretation - Categorical variable

R by default set the categorical variables using **dummy-coding**. In this case we set the reference category to `online`.

```r
fit <- glm(solved ~ class, family = poisson(link = "log"), data = dat)
summary(fit)
```

```
##
## Call:
## glm(formula = solved ~ class, family = poisson(link = "log"),
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9098  -0.7226  -0.1125   0.5905   3.3774
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.33795    0.04394  53.211  < 2e-16 ***
## classinperson  0.34853    0.05738   6.074 1.25e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 139.40  on 99  degrees of freedom
## Residual deviance: 101.95  on 98  degrees of freedom
## AIC: 538.13
##
## Number of Fisher Scoring iterations: 4
```

Parameters intepretation - Categorical variable

- Similarly to the previous example, the intercept is the expected number of solved exercises when the `class` is 0. Thus the expected number of solved exercises for online students.
- the `classinperson` is the difference in log solved exercises between online and in person classes. In the response scale is the expected increase in the ratio of solved exercises. People doing in person classes solve 141.6988417 of the exercises of people doing online classes

```
c(coef(fit)["(Intercept)"], exp(coef(fit)["(Intercept)"]))
```

```
## (Intercept) (Intercept)
##    2.337952   10.360000
```

```
c(coef(fit)["classinperson"], exp(coef(fit)["classinperson"]))
```

```
## classinperson classinperson
##     0.3485338     1.4169884
```

Overdispersion

Overdispersion

Overdispersion concerns observing a greater variance compared to what would have been expected by the model.

To understand **Overdispersion** we define the **scaled deviance** as:

$$D^* = \frac{D}{\phi}$$
$$D = -2[\log(\mathcal{L}_{current}) - (\log \mathcal{L}_{sat})]$$

Where D is the deviance that we already defined and ϕ is the dispersion parameter of the distribution. For Binomial and Poisson models $\phi = 1$ and $D = D^*$. We know that $D^* \sim \chi^2_{n-p-1}$.

Overdispersion

Without knowing ϕ, we can estimate it using:

$$\hat{\phi} = \frac{D}{n - p - 1}$$

If the model is correctly specified for binomial and poisson models the ratio is equal to 1, of the ratio is > 1 there is evidence for overdispersion. In practical terms, if the residual deviance is higher than the residuals degrees of freedom, there is evidence for overdispersion.

Overdispersion

We can compute the ratio between `fit$deviance / fit$df.residual` $= 1.0402733$:

```
##
## Call:
## glm(formula = solved ~ class, family = poisson(link = "log"),
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9098  -0.7226  -0.1125   0.5905   3.3774
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.33795    0.04394  53.211  < 2e-16 ***
## classinperson  0.34853    0.05738   6.074 1.25e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 139.40  on 99  degrees of freedom
## Residual deviance: 101.95  on 98  degrees of freedom
## AIC: 538.13
##
## Number of Fisher Scoring iterations: 4
```

Testing overdispersion

To formally test for overdispersion i.e. testing if the ratio is significantly different from 1 we can use the `performance::check_overdispersion()` function.

```
performance::check_overdispersion(fit)
```

```
## # Overdispersion test
##
##        dispersion ratio =   1.080
##   Pearson's Chi-Squared = 105.807
##                 p-value =   0.277
```

Regression and Multilevel Models, (Gelman and Hill, 2007), Ch. 6, pp. 114-115

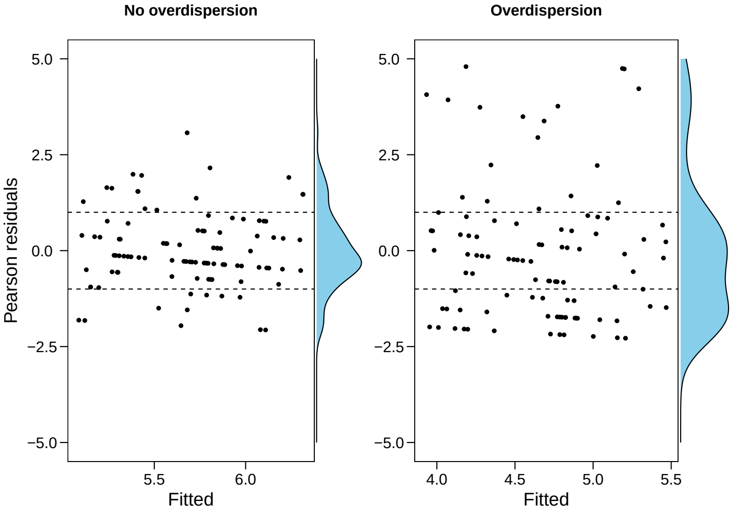
Overdispersion plot

Pearson residuals are defined as:

$$r_p = \frac{y_i - \hat{y}_i}{\sqrt{V(\hat{y}_i)}}$$
$$V(\hat{y}_i) = \sqrt{\hat{y}_i}$$

Remember that the mean and the variance are the same in Poisson models. If the model is correct, the standardized residuals should be normally distributed with mean 0 and variance 1.

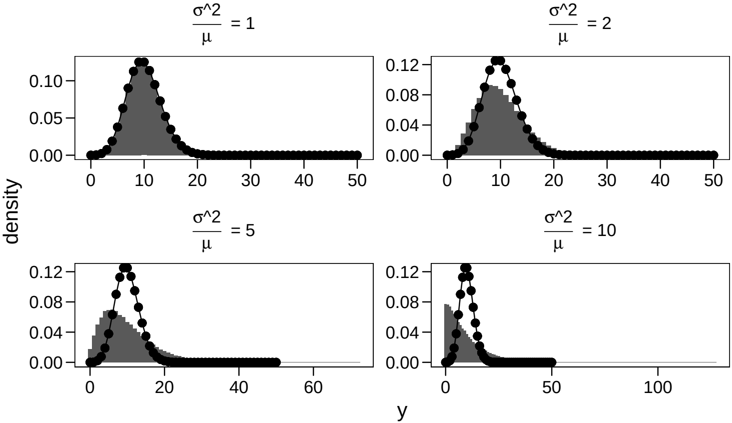
Overdispersion plot



Regression and Multilevel Models, (Gelman & Hill, 2007), Ch. 6, pp. 114

Variance-mean relationship

The overdispersion can be expressed also in terms of variance-mean ratio. In fact, when the ratio is 1, there is no evidence of overdispersion.

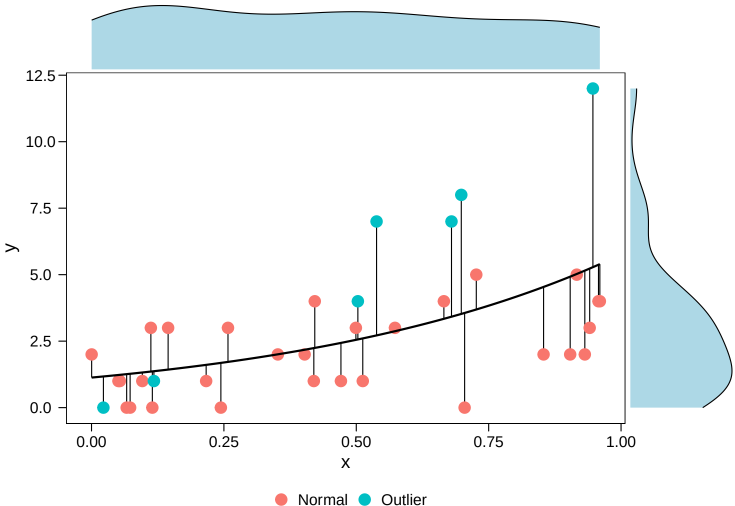


Causes of overdispersion

There could be multiple causes for overdispersion:

Outliers or anomalous data

This (simulated) dataset contains $n = 30$ observations coming from a poisson model in the form $y = 1 + 2x$ and $n = 7$ observations coming from a model $y = 1 + 10x$.



Outliers or anomalous data

Clearly the sum of squared pearson residuals is inflated by these values producing more variance compared to what should be expected.

```
c(mean = mean(datout$y), var = var(datout$y)) # mean and variance should be similar
```

```
##     mean      var
## 2.756757 6.689189
```

```
performance::check_overdispersion(fit)
```

```
## # Overdispersion test
##
##        dispersion ratio =  1.515
##   Pearson's Chi-Squared = 53.019
##                 p-value =  0.026
```

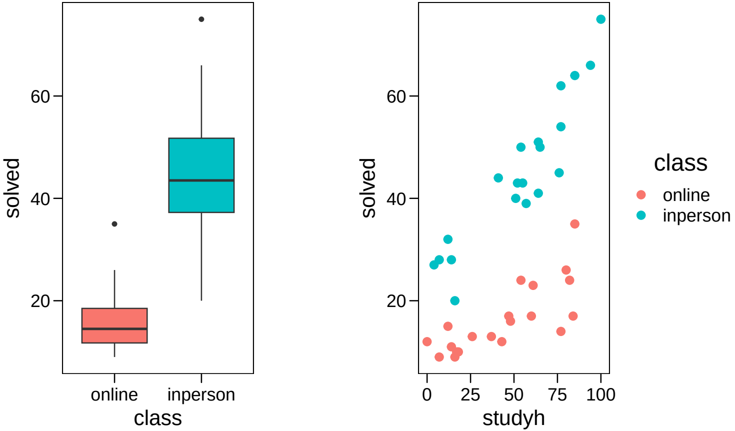
Missing important variables in the model

Let's imagine to analyze again the dataset with the number of solved exercises. We have the effect of the `studyh` variable. In addition we have the effect of the `class` variable, without interaction.

id	class	studyh	class.c	lp	solved
1	online	82	0	22.6127119305986	24
2	inperson	65	1	47.7341622043588	50
3	online	12	0	11.2682503013197	15
4	inperson	54	1	42.7852617196088	50
...	...	...	...	...	...
37	online	85	0	23.2978997148077	35
38	inperson	55	1	43.2131143368049	43
39	online	80	0	22.1671521719426	26
40	inperson	77	1	53.7880487642509	54

Missing important variables in the model

We can also have a look at the data:



Missing important variables in the model

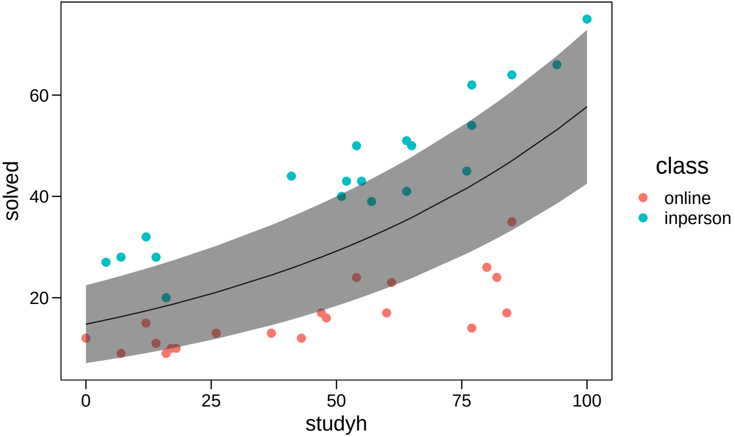
Now let's fit the model considering only `studyh` and ignoring the group:

```
fit <- glm(solved ~ studyh, data = dat, family = poisson(link = "log"))
summary(fit)
```

```
##
## Call:
## glm(formula = solved ~ studyh, family = poisson(link = "log"),
##     data = dat)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -5.047  -2.247  -0.107   2.214   3.246
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 2.692791   0.068831   39.12   <2e-16 ***
## studyh      0.013624   0.001063   12.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 417.99  on 39  degrees of freedom
## Residual deviance: 243.99  on 38  degrees of freedom
## AIC: 451.39
##
## Number of Fisher Scoring iterations: 4
```

Missing important variables in the model

Essentially, we are fitting an average relationship across groups but the model ignore that the two groups differs, thus the observed variance is definitely higher beacuse we need two separate means to explain the `class` effect.



Missing important variables in the model

By fitting the appropriate model, the overdispersion is no longer a problem:

```
fit2 <- glm(solved ~ studyh + class, data = dat, family = poisson(link = "log"))
summary(fit2)
```

```
##
## Call:
## glm(formula = solved ~ studyh + class, family = poisson(link = "log"),
##     data = dat)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.93006  -0.48422   0.00417   0.43834   1.97814
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.275159   0.078239   29.08   <2e-16 ***
## studyh        0.010895   0.001068   10.21   <2e-16 ***
## classinperson 0.907365   0.065373   13.88   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 417.993  on 39  degrees of freedom
## Residual deviance:  29.022  on 37  degrees of freedom
## AIC: 238.41
##
## Number of Fisher Scoring iterations: 4
```

Missing important variables in the model

```r
performance::check_overdispersion(fit)
```

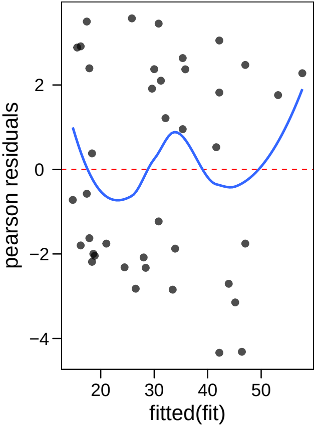
```
## # Overdispersion test
##
##        dispersion ratio =   6.115
##   Pearson's Chi-Squared = 232.369
##                 p-value = < 0.001
```

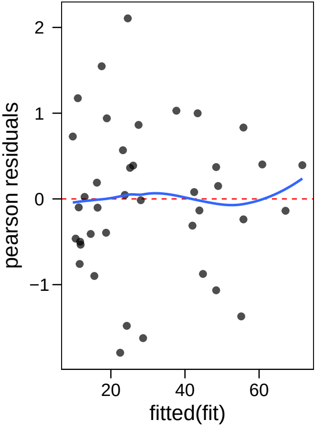
```r
performance::check_overdispersion(fit2)
```

```
## # Overdispersion test
##
##        dispersion ratio =  0.777
##   Pearson's Chi-Squared = 28.738
##                 p-value =  0.832
```

Missing important variables in the model

Also the residuals plot clearly improved after including all relevant predictors:





Why worring about overdispersion?

Before analyzing the two main strategies to deal with overdispersion, it is important to understand why it is very problematic.

Despite the estimated parameters are not affected by overdispersion, the standard error are very underestimated as a function of the degree of overdispersion.

Underestimated standard errors produce unrealistic precise parameters estimations inflating the type-1 error (i.e., parameters tends to be more significant than when overdispersion is considered)

Why worring about overdispersion?

The estimated overdispersion of `fit` is ~ 6.4208936. The `summary()` function in R has a `dispersion` argument to check how model parameters are affected.

```
phi <- fit$deviance / fit$df.residual
summary(fit, dispersion = 1)$coefficients # the default
```

```
##               Estimate  Std. Error  z value     Pr(>|z|)
## (Intercept) 2.69279058 0.068830558 39.12202 0.000000e+00
## studyh      0.01362422 0.001062724 12.82009 1.265575e-37
```

```
summary(fit, dispersion = 2)$coefficients # the default
```

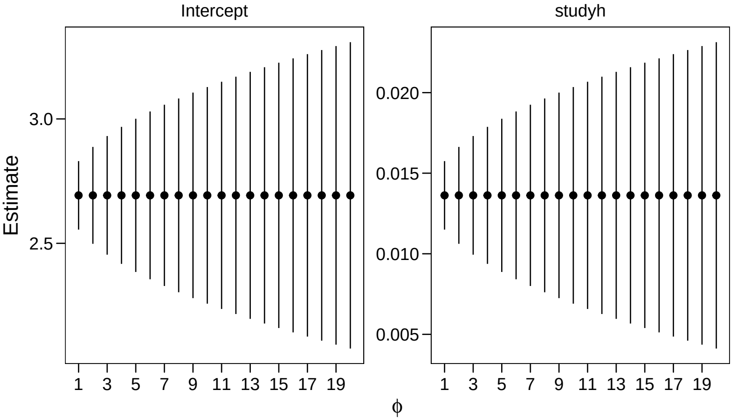
```
##               Estimate  Std. Error   z value      Pr(>|z|)
## (Intercept) 2.69279058 0.097341109 27.663447 1.923115e-168
## studyh      0.01362422 0.001502919  9.065171  1.244091e-19
```

```
summary(fit, dispersion = phi)$coefficients # the appropriate
```

```
##               Estimate  Std. Error   z value     Pr(>|z|)
## (Intercept) 2.69279058 0.174413070 15.439156 8.926080e-54
## studyh      0.01362422 0.002692888  5.059333 4.207258e-07
```

Why worring about overdispersion?

By using multiple values for ϕ we can see the impact on the standard error:



Dealing with overdispersion

Dealing with overdispersion

If all the variables are included and there are no outliers, the phenomenon itself contains more variability compared to what predicted by the Poisson. There are two main approaches to deal with the situation:

- **quasi-poisson** model
- poisson-gamma model AKA **negative-binomial model**

Quasi-poisson model

The **quasi-poisson** model is essentially a poisson model that estimate the ϕ parameter and adjust the standard errors accordingly. Again, assuming to fit the `studyh` only model (with overdispersion):

```
fit <- glm(solved ~ studyh, data = dat, family = quasipoisson(link = "log"))
summary(fit)
```

```
##
## Call:
## glm(formula = solved ~ studyh, family = quasipoisson(link = "log"),
##     data = dat)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -5.047  -2.247  -0.107   2.214   3.246
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.692791   0.170209  15.821  < 2e-16 ***
## studyh      0.013624   0.002628   5.184 7.46e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 6.115055)
##
##     Null deviance: 417.99  on 39  degrees of freedom
## Residual deviance: 243.99  on 38  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

Quasi-poisson model

The quasi-poisson model estimates the same parameter and adjust standard errors as we did on slide 39. All other parameters are the same.

The quasi-poisson model is useful because it is a very simple way to deal with overdispersion.

The variance ($V(\mu)$) of the Poisson model is no longer μ but $V(\mu) = \mu\phi$. When ϕ is close to 1, the quasi-poisson model is the same as a standard poisson model.

Problems of Quasi-* model

The main problem of quasi-* models is that they are not a specific distribution family and there is not a likelihood function. For this reason, we cannot perform model comparison the standard AIC/BIC. See @ref(Ver_Hoef2007-jc) for an overview.

Negative-binomial model

A negative binomial model is a separated random component with two parameters: the mean as in standard poisson model and the dispersion parameter. Similarly to the quasi-poisson model it estimates a dispersion parameter.

Practically the negative-binomial model is constructed from a hierarchical model:

$$
\begin{aligned}
y_i &\sim Poisson(\lambda_i) \\
\lambda_i &\sim Gamma(\mu_i, \phi)
\end{aligned}
$$

In this way, the Gamma distribution regulate the dispersion around the expected value under the poisson model.

Negative-binomial model

The mean of the negative binomial distribution is λ and the variance is $\lambda + \frac{\lambda^2}{\theta}$

$$p(y_i; \mu_i, k) = \frac{\Gamma(y_i + k)}{\Gamma(y_i + 1)\Gamma(k)} \left(\frac{\mu_i}{\mu_i + k}\right)^{y_i} \left(1 - \frac{\mu_i}{\mu_i + k}\right)^{k}$$

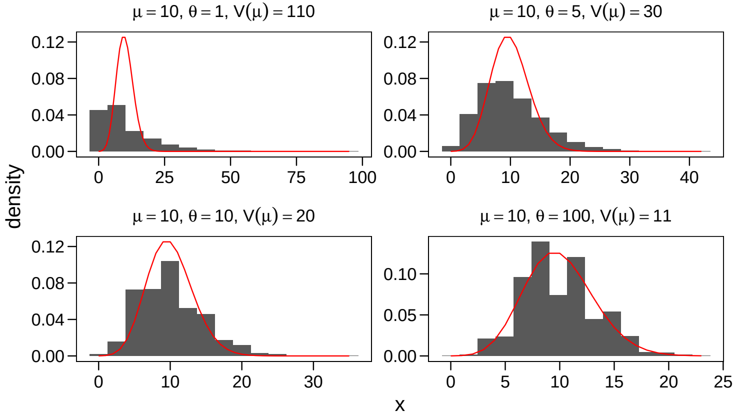
Where $k = 1/\phi$, $\Gamma()$ is the gamma function. The mean is μ_i and the variance is $\mu_i + \frac{\mu_i^2}{k}$

Negative-binomial model

Compared to the Poisson model, the negative binomial allows for overdispersion estimating the parameter ϕ and compared to the quasi-poisson model the variance is not a linear increase of the mean ($V(\mu) = \phi\mu$) but have a quadratic relationship $V(\mu) = \mu + \mu^2/\phi$

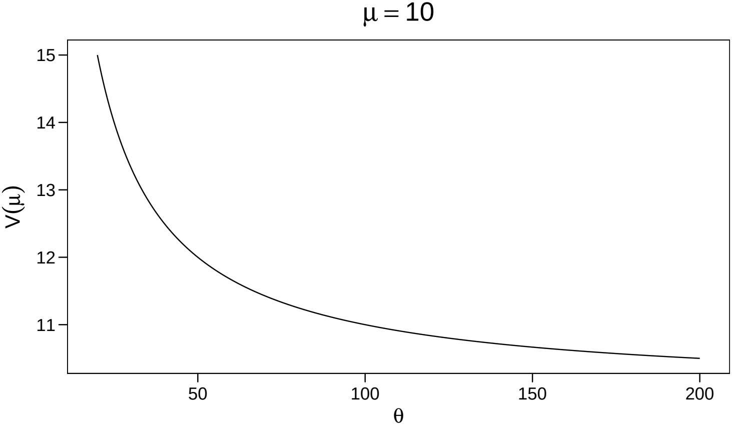
Negative-binomial model

We can use the MASS package to implement the negative binomial distribution using the `MASS::rnegbin()`:



Negative-binomial model

The θ parameter is the estimated dispersion. To note, is not the same as ϕ in the quasi-poisson model. As θ increase, the overdispersion is reduced and the model is similar to a standard Poisson model.



References

filippo.gambarota@unipd.it

github.com/filippogambarota