Instructions: Define and Plot Candidate Models

Download and source the file “plotModels.r” into R: source(“plotModels.r”)
(If you want to plot and fit a compartment model, download and source “compartfit.r”.
We allow the log-log link in GLMs with the binomial family. Therefore, the “make.link”
and “binomial” function have been slightly updated from their basic definition in R. To
get the updated version, use source(“makelink.r”) and source(“binomial.r”). You only
need to do this if you plan to use the log-log link in any of your candidate models.)

1. Define the dose levels, e.g. dose=c(0,1,4,12,24)
2. Define the candidate models: To define candidate models, we need two
   components, the model type and the family of the response, i.e., the specifications of
   a typical glm() call. By default, if no model type is specified, the linear predictor is
   assumed to be equal to a+b*dose. All possible family-link combinations supported by
   R are technically valid. All other arguments that follow are optional. E.g.,

   M1 = list(family=binomial(link=logit))

   specifies a logistic regression model with linear predictor a+b*dose, with a logit-link
   (basic logistic regression). To specify a glm with a different link, use

   M2 = list(family=binomial(link=log))

   Other families such as gaussian are also possible. Note that family=quasi offers a
   variety of different link and variance options falling within the quasi-likelihood
   framework. Specifying family=quasi(link=”identity”, variance=”mu(1-mu)”)
   fits a model with identity link and binomial random components. Various other options are
   available, such as

   list(family=quasi(link=”1/mu^2”, variance=”mu(1-mu)”))

   (Note: While models with quasi link work fine for plotting, fitting them is often
   cumbersome. Often, the data have to be entered in the “long”, i.e., binary form and
   starting values have to be supplied.)

   To specify a model with linear predictor a +b*(dose+off)^power, use the
   model=pow(dose,p) command, e.g.,

   M3=list(model=pow(dose,0.5), family=binomial())

   By default, off=0 when the power p > 0 and off=1 when the power p <= 0, but this
   can be changed via the off= option (note that off will not be a parameter estimated
   from the data, although this feature might be available in a future version). E.g.,

   M4=list(model=pow(dose,-1,off=2), family=binomial())
specifies a linear predictor of form \( a + b/(\text{dose}+2) \). When \( p=0 \), a GLM in \( \log(\text{dose}) \) is specified (linear predictor: \( a + b*\log(\text{dose}+\text{off}) \)):

\[
\text{M5} = \text{list} \left( \text{model} = \text{pow} \left( \text{dose}, 0 \right), \text{family} = \text{binomial}() \right)
\]

Models exponential in dose are specified via the \( \text{model} = \text{expo} \left( \text{dose}, \text{o} \right) \) command. E.g., \( \text{model} = \text{expo} \left( \text{dose}, 1 \right) \) specifies linear predictor \( a + b \text{exp}(\text{dose}) \), while \( \text{model} = \text{expo} \left( \text{dose}, 2 \right) \) models \( a + b \text{exp}(\text{exp}(\text{dose})) \). These two orders currently are the only two implemented. In general, both \( \text{pow}() \) and \( \text{expo}() \) allow to supply a scaling factor to scale the dose, via \( \text{scale}=\text{(defaults to 1)} \). For instance,

\[
\text{M6} = \text{list} \left( \text{model} = \text{expo} \left( \text{dose}, 2, \text{scale} = \text{max}(\text{dose}) \right), \text{family} = \text{binomial}() \right)
\]

fits the double exponential model \( a + b \text{exp}(\text{exp}(\text{dose}/\text{scale})) \), where dose is scaled by the maximum dose. As does \( \text{pow}() \), \( \text{expo}() \) also allows to supply an offset via \( \text{off}=\text{(defaults to 0 in expo())} \).

To fit a GLM with linear predictor equal to \( a + b1(\text{dose}+\text{off})^{\text{power1}} + b2(\text{dose}+\text{off})^{\text{power2}} \), such as a model quadratic in dose, use \( \text{model} = \text{pow} \left( \text{dose}, p1, p2 \right) \), as in

\[
\text{list} \left( \text{model} = \text{pow} \left( \text{dose}, 1, 2 \right), \text{family} = \text{binomial}() \right).
\]

To completely specify candidate models with three parameters, we need to give an estimate of the dose at which the efficacy peaks. (If no such estimate is provided, by default it is assumed that the maximum efficacy occurs at the highest dose.) Supplying an estimate is done via the \( \text{dmax} \) option in \( \text{pow}() \), which defaults to \( \text{dmax}=\text{max}(\text{dose}) \). E.g., specifying that the dose at which efficacy is maximized occurs at 14 for the quadratic model above, use

\[
\text{M7} = \text{list} \left( \text{model} = \text{pow} \left( \text{dose}, c(1, 2), \text{dmax}=14 \right), \text{family} = \text{binomial}() \right)
\]

Any other combination of powers \( (p1, p2) \) including 0 for log-dose is also valid, such as in

\[
\text{M8} = \text{list} \left( \text{model} = \text{pow} \left( \text{dose}, c(0, -1), \text{dmax}=8 \right), \text{family} = \text{binomial}() \right)
\]

Default offsets \( \text{off} \) equal 0 for positive powers and 1 for 0 and negative powers. These can be overwritten via \( \text{off}=\text{c(\text{off1, off2})} \); note however, that if you specify an offset, you need to specify one for both powers, even if one is just the default. E.g., if you use \( \text{model} = \text{pow} = (1, -1) \), you need to include \( \text{off}=(0, 0.5) \) to specify no offset for the dose with the positive power component and an offset of 0.5 for the dose with the negative power component.

3. **Put all models in a list:**

\[
\text{models} = \text{list} \left( \text{M1, M2, M3, M4, M5, M6, M7, M8} \right)
\]
Optionally, to name your models, you can include tags in the list as in `models=list(Logit=M, LogLin=M2,…)` or you can define labels, see below.

4. **Call the function** `plotModels(dose, models)` **to plot the candidate models specified in** `models` **over the range of dose.** Optional arguments include
   • `low=` to specify the predicted placebo effect (defaults to 0.0001)
   • `high=` to specify the predicted maximum possible efficacy (defaults to 0.9999)
   • `label=` to include labels for the models (defaults to c (“M1”, “M2”,…))
   • `steps=` step size for plotting predicted probabilities (defaults to min(dose[-1])/100)

The following code (parts in gray make the plot nicer but are not needed) produces the plot of the 8 candidate models specified above in Figure 1 when the estimated placebo effect is 30% and the estimated maximum efficacy is 65%:

```r
plotModels(dose, models, low=0.3, high=0.65, label=label)
```

![Candidate Models](image)

Figure 1: Plot of candidate models via `plotModels.bin()` function
**Remark:** plotModels.bin can also be used to plot candidate models with other family functions, such as Gaussian or gamma.

**Instructions: Fit and Select Candidate Models**

Source the file “permG2.r” into R: source(“permG2.r”)

1. We assume that you have supplied dose levels and a list of candidate models, see 1-3 above.
2. **Supply** vector of individual (binary or continuous) or aggregate responses (and sample size) at the dose levels, e.g.,

   \[ \text{resp} = c(1,0,1,0,1,\ldots,1) \]

   if you have individual observations (note that the dose vector needs to be of the same length as resp), or

   \[
   \begin{align*}
   y & = c(38,52,67,59,58) \\
   n & = c(100,102,98,99,94) \\
   \text{resp} & = (\text{cbind}(y,n-y)
   \end{align*}
   \]

   if you have aggregate data, i.e., binomial counts at each dose level. If you want to use a compartment model, you have to supply the response in this latter form.

3. Call function **permG2(dose, resp, models, perms=5000)** to obtain a (multiplicity adjusted) permutation P-value for the test of Proof of Concept with your candidate models and identify those models that yield a significant dose-response. Also, for each candidate model, the MED is estimated. Based on these, a weighted MED estimate over all models with existing MED is computed. Optional parameters to **permG2()** are

   - **clinRel** … the clinical relevant effect (defaults to 0)
   - **alpha** … the familywise error rate that you wish to control (defaults to 0.05)
   - **gamma** … the confidence level of the confidence interval for estimating the MED (defaults to 0.05)
   - **label** … labels of your models (defaults to c(“M1”, “M2”,…))
   - **trace** … an option to print out the critical value (i.e., the 1-alpha quantile of the permutation distribution of maxG2) after trace permutations. This can help monitor convergence of the permutation distribution.
   - **seed** … the random seed used to generate the permutations. Setting the seed will allow for reproducible results (defaults to .Random.seed)
   - **steps** … the size of the steps that are used in the linear search for the MED and that also control step size for plotting estimated probabilities (defaults to \text{min(dose[-1])/100})
The call `permG2(dose, resp, models, perms=0)` performs no permutations but fits all candidate models and finds asymptotic P-values, MED estimates and the weighted MED estimate.

The return value of `permG2()` is an object of type ‘maxG2’. Regular ‘print’, ‘summary’ and ‘plot’ statements work with this object.

**Example:** With the data in aggregate form and `resp` defined as above (and optional labels defined above), we let

```r
> dr = permG2(dose, resp, models, perms=5000, clinRel=0.15, label=label, trace=1000)
```

which results in the following output, showing the critical value from the permutation distribution after trace random permutations. (If `trace` is missing, `permG2` produces no output by itself.):

```r
> dr=permG2(dose,resp,models,clinRel=0.15,perms=5000,label=label, trace=1000)
Permutation 1000 : critical value= 3.894239
Permutation 2000 : critical value= 3.930934
Permutation 3000 : critical value= 3.92441
Permutation 4000 : critical value= 3.936059
Permutation 5000 : critical value= 3.93745
```

Basic results of the procedure can be obtained by typing `dr`

```r
> dr
Proof of Concept P-value: 2e-04
MED estimate for model with maximum G2: 0.69
Weighted MED estimate: 1.93764
```

The `summary()` function gives more information:

```r
> summary(dr)
Permutation analysis of maxG2:

<table>
<thead>
<tr>
<th>model</th>
<th>obsG2</th>
<th>asympt.P</th>
<th>perm.P</th>
<th>adj.P</th>
<th>MED</th>
<th>weight</th>
<th>conv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logit</td>
<td>3.679</td>
<td>1.72e-02</td>
<td>0.0090</td>
<td>0.1534</td>
<td>NA</td>
<td>0.00000</td>
<td>1</td>
</tr>
<tr>
<td>Log-linear</td>
<td>3.247</td>
<td>2.20e-02</td>
<td>0.0110</td>
<td>0.2865</td>
<td>NA</td>
<td>0.00000</td>
<td>1</td>
</tr>
<tr>
<td>Square-root</td>
<td>8.762</td>
<td>1.04e-03</td>
<td>0.0008</td>
<td>0.0024</td>
<td>12.21</td>
<td>0.01863</td>
<td>1</td>
</tr>
<tr>
<td>1/dose</td>
<td>14.639</td>
<td>4.52e-05</td>
<td>0.0002</td>
<td>0.0004</td>
<td>2.56</td>
<td>0.35191</td>
<td>1</td>
</tr>
<tr>
<td>Log-dose</td>
<td>10.532</td>
<td>4.00e-04</td>
<td>0.0004</td>
<td>0.0014</td>
<td>7.95</td>
<td>0.04515</td>
<td>1</td>
</tr>
<tr>
<td>Double-expo</td>
<td>0.935</td>
<td>8.67e-02</td>
<td>0.3943</td>
<td>0.7530</td>
<td>NA</td>
<td>0.00000</td>
<td>1</td>
</tr>
<tr>
<td>Quadratic</td>
<td>7.008</td>
<td>4.07e-03</td>
<td>0.0028</td>
<td>0.0056</td>
<td>6.77</td>
<td>0.00775</td>
<td>1</td>
</tr>
<tr>
<td>Fract. Poly</td>
<td>15.627</td>
<td>5.47e-05</td>
<td>0.0002</td>
<td>0.0002</td>
<td>0.69</td>
<td>0.57656</td>
<td>1</td>
</tr>
</tbody>
</table>

Critical value: 3.94
Proof of Concept P-value ( 5000 Permutations ): 2e-04
MED estimate for model with largest G2: 0.69
The first column lists the candidate model, the second their G2 statistic. Asymptotic p-values in the third column are based on the Chi-square approximation of G2 statistic (see manuscript for details). Permutation P-values in the fourth column are based on the permutation distribution of G2 for each model. That is, for a given model, the permutation P-value is the number of permutations that resulted in an equal or larger signed G2 statistic than the one observed for that model. (For large sample sizes, this is roughly equal to the asymptotic P-value). Adjusted P-values in the fifth column adjust for the multiple tests with the 8 candidate models via the single step-down maxT method of Westfall and Young.

The sixth column lists the estimated MED for each model, and the seventh column the (penalized) likelihood weight associated with each model. Finally, the last column indicates the proportion of permutations for which the model converges. (Since permuting the data, it may happen that for some permutations the glm.fit() routine doesn’t converge, or, for models with non-linear predictor, maximization is not possible.)

To obtain the histogram of the distribution of max G2, type hist(dr)

> hist(dr)

![Histogram of max G2](image)

To obtain a plot of the most significant fitted model, type plot(dr)

> plot(dr)
Dotted lines represent pointwise lower and upper confidence limits (option se.fit=FALSE suppresses plotting these). The dashed line represents the clinical relevant effect as specified in permG2(). Grey points show the sample proportions and the bid X indicates the location of the MED.

Several models can be plotted by using the which.model option in plot(). E.g., to plot the three most significant models, use

> plot(dr,which.model=c(8,4,5),se.fit=F)
There is also a trellis version, but it is not very developed yet. Type `plot1.maxG2(dr)` to get

> plot1.maxG2(dr)

![Fitted Models]

Here, the dashed line indicates the clinical relevant effect.

Type `plot1.maxG2(dr,which.model=c(8,4,5,3))` to get a panel plot of the fitted models specified in `which.model`.

> plot1.maxG2(dr,which.model=c(8,4,5,3))
The fits of candidate models can be accessed via `dr$fits$(model label)`. If no labels were assigned, then the model labels are M1, M2, .... For instance, to get information on the fit of the fractional polynomial model, type

```r
> model8=dr$fits$"Fract. Poly."
> summary(model8)
```

Call:

```r
glm(formula = resp ~ ., family = M$family, data =
as.data.frame(M$model))
```

Deviance Residuals:

```
   1       2       3       4       5
 0.2467  -0.8123  1.1330  -0.6732  0.1683
```

Coefficients:

```
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                  1.2645     0.5767   2.193  0.02833 *
log.dose                   -0.2333     0.2067  -1.129  0.25906
`dose^-1`                  -1.8050     0.6821  -2.646  0.00814 **
```

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for binomial family taken to be 1)

Null deviance: 22.1126 on 4 degrees of freedom
Residual deviance: 2.4859 on 2 degrees of freedom
AIC: 33.426

Number of Fisher Scoring iterations: 3

Since this is a GLM object, all functions that work with GLM work with it, such as

> fitted(model8)
  1         2         3         4         5
0.3680761 0.5499061 0.6290148 0.6288284 0.6085585

> residuals(model8)
 1          2          3          4          5
0.2466978 -0.8123454  1.1329738 -0.6731565  0.1683307

Type names(dr) to see various other results and input parameters accessible for the maxG2 object.

> names(dr)
[1] "model" "obsG2" "maxG2" "crit.value" "perms"
[11] "wMED" "weight" "conv" "fits" "dose"
[16] "clinRel" "gamma"