**Generalized Linear Mixed Models in SAS**

This module starts with an explanation of generalized linear models, followed by some remarks about procedures for dealing with repeated measurements with generalized models in various stats packages, then step-by-step instructions for various analyses. The first analysis is a simple example of log-hazards and logistic regression, where there is no repeated measurement and where the dependent variable is a binary outcome in a controlled trial: injured/not injured, using *log-hazards regression*, or selected/not selected, using *logistic regression*. The other analyses are aimed at investigating the relationship between a fitness test and subsequent values of performance indicator in a short season of rugby union; these data were kindly provided by Josh Darrall-Jones from one of his PhD studies at Leeds Beckett University. The performance indicator is the count of effective rucks of each player in each of up to six games, which we will analyze first as a count using *Poisson regression*, then as a proportion of the rucks attempted by the player using *logistic regression*.

The programs developed below are available in the Word files **binary outcome programs SAS.docx** and **Poisson and logistic repeated measures programs SAS.docx.** They can also be opened directly into SAS via **binary outcome programs in SAS format.sas** and **Poisson and logistic repeated measures programs in SAS format.sas**.

[Generalized Linear Models Explained](#_Generalized_Linear_Models)  
[Inferences with Generalized Linear Mixed Modeling](#_Inferences_with_Generalized)  
[Procedures for Generalized Mixed Modeling](#_Procedures_for_Generalized_1)  
[Simple Log-hazards and Logistic Regression](#_Simple_Log-hazards_and)  
[Poisson Regression with Repeated Measurement](#_Poisson_Regression_with)  
[Logistic Regression with Repeated Measurement](#_Logistic_Regression_with)

# Generalized Linear Models Explained

The *general* in *general linear (mixed) models* refers to the usual continuous numeric dependent variables we've been analyzing so far, whereas *generalized* refers to these and any other kind of dependent variable. The other kinds we have to worry about are binary variables, counts, and proportions. Binary variables are usually coded as 0 or 1 to indicate whether something occurred or was present; counts are non-negative integers representing how many things occurred or were present; proportions have values between 0 and 1, indicating the count of occurrences or presences expressed as a fraction of the total possible. If you apply the usual linear models to these variables, you can get into serious trouble: non-uniformity of errors and effects, and impossible predicted values and/or confidence limits thereof (negative counts, proportions outside the range 0 to 1). The solution is to model a transformation of the dependent variable that can go from -∞ to ∞, and you trust that the resulting effects on the transformed variable are realistic; they are likely to be more realistic than effects on the untransformed variable.

*Poisson Regression*

For counts the transformation is simply the (natural) log: a count goes from 0 to ∞, so log(count) goes from -∞ to ∞. You don't take logs of the counts, though, because if there are any zeros, you would get -∞, which becomes a missing value. What actually happens is that the model predicts the log of the count, using the observed counts, which *can* include zeros. The model also takes into account the fact that the sampling variation of a count depends on the expected (predicted or true) value of the count. For example, if you expect a count of 25, you will observe counts like 19, 27, 24, 18, 31,… The scatter of the counts is called the sampling distribution, and for counts the distribution is known as a Poisson distribution. The variance of a Poisson distribution is the same as the mean, so the standard deviation in this example is √(variance) = √25 = 5. Hence those typical numbers shown.

A linear model for a count is called Poisson regression, but what's being predicted by the linear model is the log of the count. Hence effects, which are differences or changes in a linear model, become ratios (factors) after back transformation. For example, the effect of an intervention could be an increase in the count of effective rucks in games of football; the counts after the intervention divided by the counts before might be a ratio or factor of 1.18, or an 18% increase.

Often the variance of a count is greater than the mean count, usually because the events making up the count occur non-randomly, i.e., in clusters within or between subjects. (You have to really think hard to figure out why the variance is greater than the mean, when the counts come in clusters.) The distribution of counts is then said to be an over-dispersed Poisson. Occasionally counts can even be under-dispersed. It's important to allow for the dispersion to be different from that of a pure Poisson, because it makes a difference to the estimates and their confidence limits. It also makes a difference to standardized effects, because the overdispersion affects the observed between-subject standard deviation.

*Logistic Regression*

For the analysis of the proportion of something, the transformation has to turn the lowest possible proportion (0) into -∞ and the highest possible proportion (1) into ∞. The logistic or log-odds transformation does the job: the odds of a proportion p is p/(1-p), which goes from 0 to ∞, so the log of the odds goes from -∞ to ∞.

A linear model predicting the log of the odds is called logistic regression. As with Poisson regression, the analytic procedure does the transformation, so you can input observed proportions of zero or 1. You can also input occurrences coded as 0 or 1, and the procedure aggregates these into proportions. Effects in the linear model are differences or changes in the log of the odds, so these turn into odds ratios after back-transformation. Odds ratios are the same as proportion ratios, when proportions are small (<0.1 or <10%): p1/(1-p1) is approximately p1, p2/(1-p2) is approximately p2, so the odds ratio p2/(1-p2)/[p1/(1-p1)] is approximately p2/p1. So that's OK when proportion ratios are what need to be evaluated for magnitude and uncertainty in the magnitude. However, when the proportions are not small, you have to use a reference proportion to convert the odds ratios back into proportion ratios so that you can make sense of the effect. For example, if the odds ratio for a treatment effect is 2.0, and the reference (control-group) proportion is 0.30 (30%), the reference odds is 0.3/0.7, so the odds in the experimental group is 2.0\*0.3/0.7 = 0.6/0.7 = 0.857, so the experimental proportion is given by p/(1-p) = 0.857, from which it follows that p = 1/(1+0.857) = 0.539, so (finally!) the effect expressed as a proportion ratio is 0.539/0.30 = 1.80, which is a bit less than the original odds ratio of 2.0. If the reference proportion had been 0.70, say, the resulting proportion ratio would be only 1.18, which is now considerably less than the odds ratio. Moral: don't interpret odds ratios as proportion ratios unless the proportions are small.

A special application of logistic regression is the analysis of win-lose outcomes in match-play sports. Here it's best to convert the odds ratio to a proportion difference rather than a proportion ratio, and you center the proportion difference on 0.5, or 50%. For example, if the odds ratio is 2.0, the two proportions are 0.41 and 0.59 (via algebra), or 41% and 59%, so the proportion difference is 0.18 or 18%. In other words, the effect will take the athlete or the team from winning 4.1 matches in every 10 matches up to 5.9 matches in every 10 matches, an increase of 1.8 in every 10 matches. This way of representing the outcome–the change in the number of matches won against an otherwise nearly equal opponent–allows the magnitude to be interpreted directly using my scale for competitive athletes: one competition in every 10 is small, and 3, 5, 7 and 9 in every 10 are moderate, large, very large and extremely large, respectively. In this example the effect would be small, but of course you would have to interpret the confidence limits to decide whether the effect was clear.

Proportions are made up of counts, so they too can be over- or under-dispersed. Substantial over- and under-dispersion occurs when analyzing time spent in an activity as a proportion of the whole time, where the time in seconds or minutes is treated as a count. Obviously each second or minute of the activity is not an independent event! So again, it's important to allow dispersion to be different from that expected for the odds of independent events (which is given by the binomial distribution). The stats procedure takes care of the details.

*Log-hazards Regression*

When modeling the presence or absence of something with a value of 0 or 1, logistic regression works fine, but if the presence or absence is a time-dependent event, such as an injury, it's not appropriate to model the log of the odds. Instead you model the log of the probability that the event will occur in subjects who are as yet unaffected. To understand why, think: if the chance of getting injured in one day is whatever really low value in a reference group or condition, the chance of getting injured in another group will also be really small, but it could be higher or lower by some reasonable factor, such as 1.53 (a 53% greater chance of injury). This short-term or instantaneous risk ratio is called the incidence-rate ratio or the hazard ratio. Over a period of two days, the chance of getting injured will be twice as high in both groups, but the risk ratio will still be 1.53. But if you consider a much longer period, you end up with substantial proportions injured in the two groups, and eventually almost everyone would be injured in both groups. By this stage the risk ratio drops to nearly 1.0 and the odds ratio becomes huge, but the incidence-rate ratio for those who still aren't injured could still be 1.53. It would therefore be wrong to model the log of odds, because the odds ratio changes with the proportion, and therefore effects change with time.

The solution is to use a transformation that converts probabilities to hazards rather than odds. The transformation is called the complementary log-log = log(‑log(1-p)), where p is the proportion affected), and the sampling distribution is still the binomial. The resulting linear model does not have an official name that I can find, so I've called it *log-hazards regression*. It's based on the assumption that the hazard ratio doesn't change with changing proportions, but of course the same kind of assumption applies to logistic regression (the odds ratio doesn't change with increasing proportions) and Poisson regression (the count ratio doesn't change with increasing counts). A more sophisticated version of this kind of analysis is available as something called proportional-hazards or Cox regression, in which the time to the event rather than the occurrence of the event is modeled. Not used here!

*Cumulative Logistic Regression*

If you are modeling outcomes in a sport where draws (ties) are common, cumulative logistic regression is the way to go. In this model, the odds ratio for winning or drawing vs losing is assumed to be the same as for winning vs drawing or losing. The transformation is called the cumulative logit, and the sampling distribution is the multinomial. The odds ratio needs to be converted to a proportion difference for interpretation.

Likert scales with only a few levels are supposed to be analyzed with cumulative logistic regression, but interpreting the magnitude of the effects is a nightmare. I usually invoke the central-limit theorem and analyze the Likert scores with the usual mixed linear model. I have also justified this approach by comparing the t statistics provided by both methods, to show that they give the same answer. It is much easier to do magnitude-based inference with differences or changes in Likert scores than with cumulative odds ratios.

# Inferences with Generalized Linear Mixed Modeling

There are three challenges: magnitude thresholds, confidence limits, and probabilities for magnitude-based inference.

**Magnitude thresholds**. For counts, proportions and hazards, the fixed effects and standard deviations are expressed as ratios, with default magnitude thresholds for small, moderate, large, very large and extremely large fixed effects of 0.9, 0.7, 0.5, 0.3 and 0.1 for reductions and their inverses for increases, 1.11, 1.43, 2.0, 3.3 and 10. Thresholds for SDs expressed as ratios are the square root of those for fixed effects, equivalent to half the thresholds in log units. Effects for proportions representing matches won can be expressed as a proportion difference and then as extra matches per 10 otherwise evenly matched matches, in which case the thresholds are ±1, ±3, ±5, ±7 and ±9 for fixed effects and half these values for SDs. For counts, and for proportions representing values for individuals, it is also possible to define thresholds via standardization (by dividing the mean effect by a between-subject SD), with thresholds of ±0.20, ±0.60, ±1.20, ±2.0 and ±4.0 for fixed effects and half these values of SDs. Most of the time the *observed* between-subject SD should be used for standardizing, and that SD should be derived from the model by adding the Poisson or binomial sampling variance (including any over-dispersion) with any pure between-subject variance derived from the random effects. It all has to be done with the log-transformed variances, but fortunately there is a simple formula for the log of the (over-dispersed) sampling variance for Poisson and binomial distributions. These formulae are incorporated into the spreadsheets for processing the results from SAS Studio.

**Confidence limits**. Hazards and counts, and confidence limits for their ratios, are given directly by SAS. However, when the dependent is a proportion, SAS provides only odds and confidence limits for their ratios. I have already explained above how the proportion ratio is derived from the odds ratio using a reference proportion, but I write code to output both proportions making up an effect (the mean values of the two levels of a nominal effect, or the predicted mean values for the mean – 1SD and the mean + 1SD of a linear numeric effect), so you can see what the proportion ratio represents. Confidence limits for the proportion ratio are more of a challenge. Applying the confidence limits for the odds ratio to the reference odds, then converting to proportions, doesn't work well. Instead you have to apply the t statistic or p value for the odds ratio (or in the case of SDs, the z score for the variance) to the log of the proportion ratio, a trick that is based on the reasonable assumption of a log-normal sampling distribution for the proportion ratio (it goes from -∞ to ∞). I have done simulations to check that it works. The proportion ratio and its confidence limits expressed in standardized units are simply (!) the log of the odds ratio and its confidence limits divided by the between-subject SD expressed as the log of an odds ratio. Confidence limits for extra matches won are calculated directly from the confidence limits for the odds ratio.

**Probabilities** representing *possible*, *likely* and so on for magnitude-based inference come from the same assumptions of the sampling distributions used to derive the confidence limits. The smallest important increase and decrease for counts and proportion ratios based on standardization have to be worked out for each effect. Everything is done for you in the spreadsheets!

# Procedures for Generalized Linear Mixed Modeling

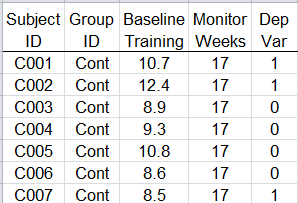
There are two approaches to generalized linear mixed models in SAS: generalized estimating equations (GEEs) (via Proc Genmod) and generalized linear models (via Proc Glimmix). GEEs were the first extension of mixed models to repeated measurements of counts and proportions, and the approach is limited to random effects representing repeated measurements on subjects. The other approach is more advanced and allows other random effects (e.g., including games as a randomly sampled variable). I now use only Glimmix. SPSS is still developing the Glimmix approach, and what they've got in their Version 23 is useless. I don't know about R, but I doubt if it even comes close to SAS.

Interestingly, if you choose Generalized Linear Models in Tasks/Statistics, you are given Proc Genmod, so we will have to use my code from the outset.

Disclaimer: I don't understand a lot of the jargon in the documentation for Proc Glimmix. Maybe this part of the workshop needs to be presented by a well-qualified statistician, which I am not.

# Simple Log-hazards and Logistic Regression

1. Open **binary outcome hazards or odds data.xlsx** in Excel and view the data in the first spreadsheet (**binary hazards data**):



The second spreadsheet (**binary hazards data+graph**) shows the same data graphed in several ways, while the third is an active simulation that generates a new dataset every time it is opened or modified. The fourth sheet is not used here; it shows an active simulation for generating data modeled with odds, for any experts who want to know how to do it.

The data represent a controlled trial, in which an experimental group of athletes experienced some kind of intervention. The binary dependent variable DepVar could represent a time-dependent event, such as the occurrence of injury (1) or not (0), during a subsequent period of monitoring shown by the variable MonitorWeeks. The analysis will therefore need to be log-hazard regression to compare the incidence rate (hazard) of injuries in the two groups as an incidence-rate (hazard) ratio. The modifying effect of BaselineTraining (in hours per week) on incidence rate in the two groups can also be estimated and compared as hazard ratios.

DepVar could also represent a time-independent classification, such as success (1) of failure (0) in some subsequent test or in selection in a draft for a team sport. More on that shortly. First, log-hazard regression.

1. Import the data into SAS Studio, do a proc print in Program 1 to check, and then copy this program into Program 1:

data bindata;

set import;

BaseTrainStd=BaselineTraining;

LnMonitorWeeks=log(MonitorWeeks);

proc standard data=bindata out=bindata1 mean=0 std=0.5;

var BaseTrainStd;

run;

The data step creates a copy of BaselineTraining called BaseTrainStd which is then standardized to a mean of 0 and SD of 0.5 in the proc standard step. LnMonitorWeeks will be needed as an "offset" in the log-hazard regression.

1. Check the LOG for errors, then copy this program into Program 1. Read the explanation below before running it. I have made the font size smaller for the estimate statements so some of the lines don't roll over:

ods noproctitle;

ods graphics / imagemap=on;

title "Log-hazard regression, no repeated measurement";

proc glimmix data=bindata1;

class GroupID;

model DepVar=GroupID GroupID\*BaseTrainStd/link=cloglog dist=binomial offset=LnMonitorWeeks;

estimate "Overall mean" int 1 GroupID 0.5 0.5 /cl exp alpha=0.1;

estimate "";

estimate "Control mean" int 1 GroupID 1 0/cl exp alpha=0.1;

estimate "Exptal mean " int 1 GroupID 0 1/cl exp alpha=0.1;

estimate "Exptal/Control mean" GroupID -1 1/cl exp alpha=0.1;

estimate "";

estimate "Control mean @ -1SD BaseTrain" int 1 GroupID 1 0 GroupID\*BaseTrainStd -0.5 0/cl exp alpha=0.1;

estimate "Control mean @ +1SD BaseTrain" int 1 GroupID 1 0 GroupID\*BaseTrainStd 0.5 0/cl exp alpha=0.1;

estimate "Control +1SD/-1SD BaseTrain" GroupID\*BaseTrainStd 1 0/cl exp alpha=0.1;

estimate "";

estimate "Exptal mean @ -1SD BaseTrain" int 1 GroupID 0 1 GroupID\*BaseTrainStd 0 -0.5/cl exp alpha=0.1;

estimate "Exptal mean @ +1SD BaseTrain" int 1 GroupID 0 1 GroupID\*BaseTrainStd 0 0.5/cl exp alpha=0.1;

estimate "Exptal +1SD/-1SD BaseTrain" GroupID\*BaseTrainStd 0 1/cl exp alpha=0.1;

estimate "";

estimate "Exptal/Control mean BaseTrain reference" int 1 GroupID 0.5 0.5 GroupID\*BaseTrainStd -0.5 -0.5

/cl exp alpha=0.1;

estimate "blank";

estimate "Exptal/Control +1SD/-1SD BaseTrain" GroupID\*BaseTrainStd -1 1/cl exp alpha=0.1;

lsmeans GroupID/diff cl alpha=0.1;

run;

The **proc glimmix** statement has no code for residual plots, which aren't helpful with this simple model.

The **model** statement allows for means for the two levels of GroupID, and different modifying effects of baseline training in the two groups. **link=cloglog** specifies the complementary log-log link, which is what you have to use for time-dependent binary dependent variables. **dist=binomial** is obvious, and **offset=LnMonitorWeeks** adjusts for the different periods of monitoring for individual subjects. Beware: it is very easy to forget to log-transform the offset variable!

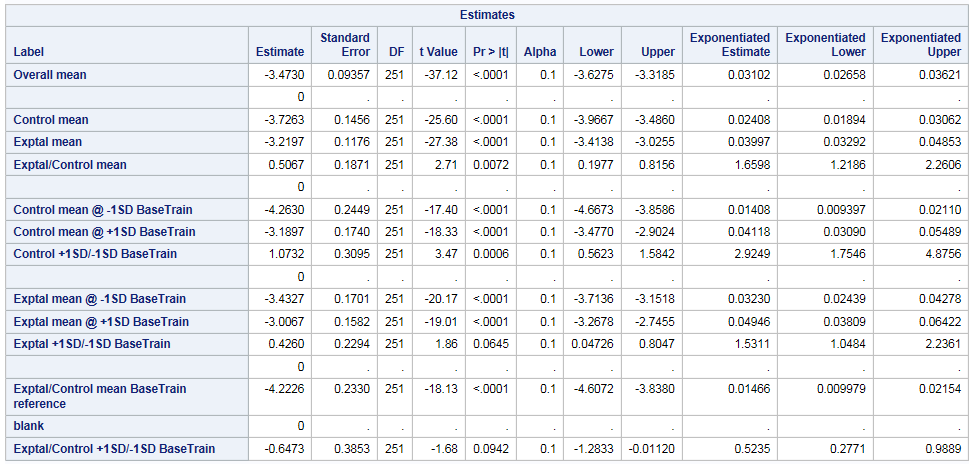
The **estimate** statement for the overall mean hazard is not particularly useful. It's there for practice.

I hope the **estimate** statements for the control and exptal mean hazards are obvious, and also their comparison, which is a ratio, because the link function is a log (of –log(1-p), but don't worry, it's right).   
**I strongly advise you to do all your estimates for effects in the manner shown: two estimates and then their comparison. This approach will help you understand the effects here, and (coming up) it's the only practical way to process proportions and to standardize effects involving counts and proportions.**

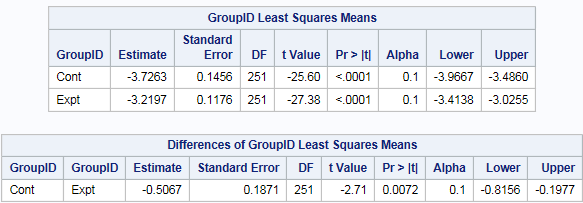
The remaining **estimate** statements are aimed at the modifying effects of baseline training. They show estimates of the hazard at 1SD below and above the mean of baseline training, followed by the ratio of the mean hazards, which is the effect of 2SD of baseline training. With log-hazards regression, you don't actually need the two means to get the effect of 2SD, but I have included the two means here so you can see what the effect of 2SD actually means: the hazard 1SD above the mean (or any arbitrary value) divided by the hazard 1SD below the mean. When we get to logistic regression, we *will* need the two means, or at least the mean at -1SD, which is another good reason for including the means here.

The **lsmeans** statement is included as a check on the estimate statements. It does not allow **exp** as an option, but it does allow something called **ilink**, which stands for *inverse link*. If you use this option, you get estimates back-transformed to proportions (per week). These will be slightly less than the hazards (which we will see are only ~0.02 and ~0.04 per week), but they are no use to us. (The **ilink** option is also available in **estimate** statements, but again, it's useless here.) The **diff** option provides differences between the least-squares means, but again, it's just a check, and there is no option for back-transformation.

1. Now highlight the block of proc glimmix code and run it. The Estimates are all that matter:



Now's a good time to check on the Estimates for the least-squares means and their difference:



Yep, all OK: the same values as in the Estimates for the control and exptal groups and their difference.

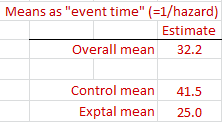
Now open the workbook **process simple binary log-hazard & logistic regression.xlsx**, check that it opens on the tab Hazards, and you will see how these results are processed.

The first thing you will see is this:

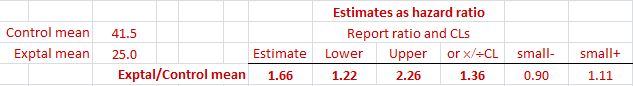


Delete an increase, so that the spreadsheet will show MBIs appropriate for a decrease in the hazard ratio being beneficial.

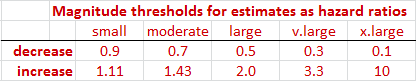
Lower down you will see that the above table of estimates has been pasted in, and to the right of it there is further processing, starting with this:



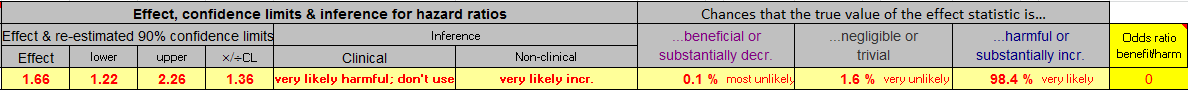
The units of hazards are hard to understand, but 1/hazard is a time constant or "event time", which is the time (here in days) it takes for a bit more than half the subjects (0.63, or 1-1/e) to experience the event. The event time for the control group is 41.3 days, while that for the experimental group is 25.0 days. Evidently the treatment increased the risk of injury, which is summarized by the hazard ratio:



The ratio of 1.66 is for the hazards, but of course it is also 41.5/25.0. The estimate and confidence limits in red are simply reproduced from the Exponentiated values in the SAS output, to make it easier to read them. The small- and small+ values are copied from a table a few rows above…

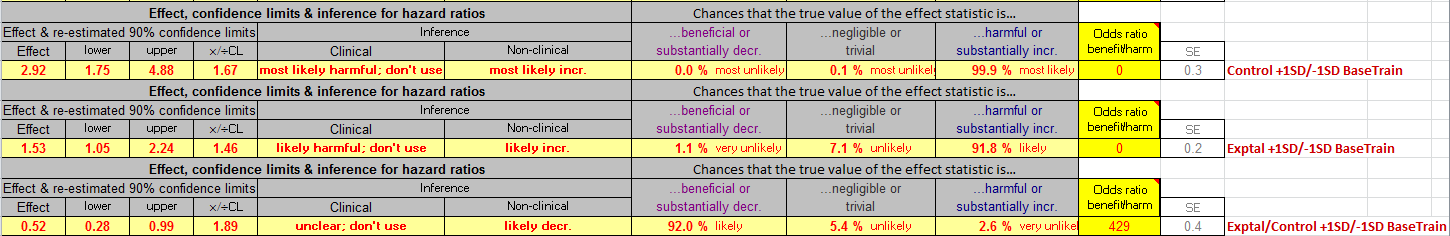


…which shows the default magnitude thresholds for hazard ratios; these allow you to assess the magnitude of the observed estimate and the confidence limits. As you can see, there's a moderate increase in risk of injury, and it's clear from the perspective of a 90% confidence interval (lower and upper limits small to large). This effect requires clinical inference, which implicitly involves consideration of the 99% confidence limit on the harm side. It's clearly harmful, but the full MBI is shown over on the right:



As it states above these cells, "they were adapted from the spreadsheet **Combine/compare effects** at Sportscience." You can modify the magnitude thresholds and confidence limits, and even the chances defining *likely*, *very unlikely* and so on.

1. Now scroll to the left to see the processing of the modifying effects of baseline training, and scroll back to the right to view the MBIs:



Covariates need to be assessed non-clinically, so these cells show that more baseline training in the control group was associated with a clear large most likely substantial increase in injury risk, whereas the effect was a clear moderate increase in the experimental group. The comparison of the effects was clear and moderate.

1. Why is there no assessment of magnitude via standardization? Because with a binary outcome variable and no repeated measurement, it is not possible to derive a meaningful between-subject SD: the subjects are either injured or not. For the same reason it is not possible to derive an SD representing individual responses, but it is nevertheless possible to estimate modifying effects of subject characteristics on risk, as we have done here with baseline training. So there are indeed individual differences in risk, but we can't express them as an SD until we have between-subject SDs in treatment and control groups.
2. Now let's pretend the data represent a time-independent classification, such as success (1) of failure (0) in a test or selection in a draft for a team sport after the intervention. In this case the aim of the analysis is to compare the proportion of successes in the experimental and control groups. The analysis will therefore need to be logistic regression. BaselineTraining can still be a modifying covariate, but MonitorWeeks is not relevant. The odds and odds ratios will need to be converted to proportions and proportion ratios. I turned on Track changes in Word so you can see the changes in the program:

title "Logistic regression, no repeated measurement";

proc glimmix data=bindata1;

class GroupID;

model DepVar=GroupID GroupID\*BaseTrainStd/link=logit dist=binomial;

estimate "Overall mean" int 1 GroupID 0.5 0.5 /cl exp alpha=0.1;

estimate "";

estimate "Control mean" int 1 GroupID 1 0/cl exp alpha=0.1;

estimate "Exptal mean" int 1 GroupID 0 1/cl exp alpha=0.1;

estimate "Exptal/Control mean" GroupID -1 1/cl exp alpha=0.1;

estimate "";

estimate "Control mean @ -1SD BaseTrain" int 1 GroupID 1 0 GroupID\*BaseTrainStd -0.5 0/cl exp alpha=0.1;

estimate "Control mean @ +1SD BaseTrain" int 1 GroupID 1 0 GroupID\*BaseTrainStd 0.5 0/cl exp alpha=0.1;

estimate "Control +1SD/-1SD BaseTrain" GroupID\*BaseTrainStd 1 0/cl exp alpha=0.1;

estimate "";

estimate "Exptal mean @ -1SD BaseTrain" int 1 GroupID 0 1 GroupID\*BaseTrainStd 0 -0.5/cl exp alpha=0.1;

estimate "Exptal mean @ +1SD BaseTrain" int 1 GroupID 0 1 GroupID\*BaseTrainStd 0 0.5/cl exp alpha=0.1;

estimate "Exptal +1SD/-1SD BaseTrain" GroupID\*BaseTrainStd 0 1/cl exp alpha=0.1;

estimate "";

estimate "Exptal/Control mean BaseTrain reference" int 1 GroupID 0.5 0.5

GroupID\*BaseTrainStd -0.5 -0.5/cl exp alpha=0.1;

estimate "blank";

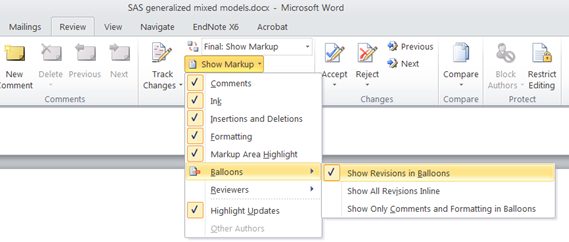
estimate "Exptal/Control +1SD/-1SD BaseTrain" GroupID\*BaseTrainStd -1 1/cl exp alpha=0.1;

lsmeans GroupID/diff cl alpha=0.1;

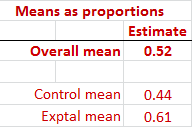
run;

There are remarkably few changes: just the link function (from cloglog to logit) and deletion of the offset. However, the processing of the output is more complicated, because the odds and odds ratios have to be converted to proportions and proportion ratios.

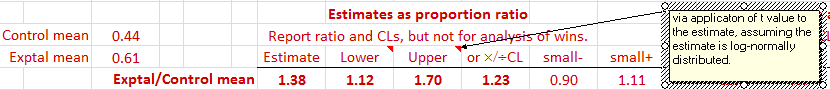
Before you copy and run the above program, make sure you have set the option for Review/Show Markup/Balloons/Show Revisions in Balloons in Word:



If instead you show changes crossed out in the text itself, the crossed-out text is submitted to SAS and the program won't run. This is how I normally write programs for the full SAS package. I use Word as an editor and submit the program to SAS via the clipboard. Tracking changes is really helpful when you are modifying programs.

Once again only the Estimates matter. Click the Proportions tab in the spreadsheet, and delete a decrease, because an increase in the odds or proportion of successes is now beneficial. You can see that I have pasted the Estimates into the spreadsheet, and on their right is the start of the processing:  


Click on the 0.52 to see the formula for converting odds to proportions: odds/(1+odds). (Specifying ilink in the estimate and lsmeans statements could have given us these proportions directly. I didn’t include it, because it gives nonsense for the Exptal/Control effect, and you can't specify exp and ilink together.)

The cells nearby show the proportion ratio and its confidence limits:  
Once again the intervention is obviously clear and beneficial, but more importantly, where did the Estimate and the Lower and Upper confidence limits come from? With hazard ratios, you can use the values that SAS provides, but odds ratios need special attention to convert them to proportion ratios. First the estimate itself has to come from the individual proportions: proportion ratio = 0.61/0.44 = 1.38. The confidence limits are then derived by assuming that the t statistic (or p value) for the odds ratio also applies to the proportion ratio. Hover the cursor over Lower or Upper to see the comment shown above.

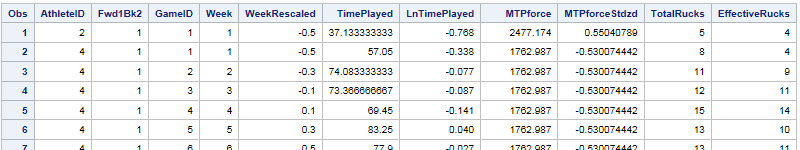
I used to derive the confidence limits by applying the confidence limits for the odds ratio to the odds of the reference group, then by transforming the resulting odds to a proportion. I checked that method with simulations and found that it did not perform well.

Scroll to the right to see the magnitude-based inferences for the treatment effect and the modifying effects of the covariate.

1. Program 1 has been saved for you already as **binary outcome programs in SAS format.sas**. Close all the tabs in SAS without saving anything and Sign Out, so we can start the next analyses with a clean screen.

# Poisson Regression with Repeated Measurement

1. We will investigate the linear relationship between a fitness test (mid-thigh pull force) and a performance indictor in games (effective rucks) across six games. We'll also see if there's a linear trend in performance over the six games.
2. Find Mixed-model Workshop/Generalized linear models in Server Files and Folders. Right-click/download the Excel file **rugby performance indicators example data.xlsx** to view it as a spreadsheet. Then double-click it to import it, run the import program, then proc print it in Program 1. You should see this:



**AthleteID** is obvious. **Fwd1Bk2** identifies forwards=1 and backs=2.

**GameID** and **Week** have the same values. We'll need both.

**WeekRescaled** goes from -0.5 to 0.5. Check cells in the Excel spreadsheet for the formula.

**TimePlayed** is in minutes. **LnTimePlayed** is the natural log of the fraction of the time expressed as a fraction of the usual game duration, 80 min. The exact duration of each game was sometimes >80 min, hence time played is >80 minutes sometimes, and LnTimePlayed is then >1.0. Check cells for the formula.

**MTPforce** is the only fitness test included here, the isometric force in a mid-thigh pull, because it shows interesting relationships with the game performance indicators.

**MTPforceStdzd** is MTPforce rescaled to give a mean of zero and a standard deviation of 0.5 for each of the two player positions, forwards and backs. Why? We'll see. Check the cells for the formulae.

**TotalRucks** and **EffectiveRucks** are each player's total rucks and effective rucks.

1. Let's generate some simple statistics for the variables that will be in the model with this bit of code:

title "Simple stats";

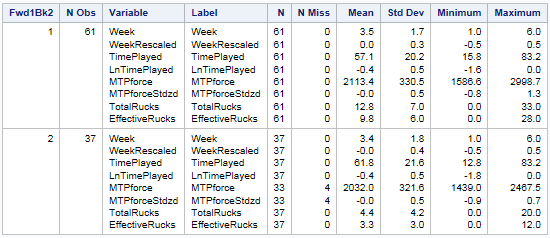
proc means n nmiss mean std min max maxdec=1 data=import;

var Week--EffectiveRucks;

class Fwd1Bk2;

run;

Note the use of -- to span a range of variables in the var statement. If Word changes -- to –, change it back to --. Copy the above code into Program 1, run it, and you will get this:



Check that WeekRescaled and MTPforceStdzd have values consistent with what I have said above. The values of the offset variable LnTimePlayed are hard to interpret, but if you multiply by 100, you get approximate percent differences from 80 min (-0.4 means approx. -40%, i.e., 40% less than 80 min).

1. Here's the Glimmix code:

ods noproctitle;

ods graphics / imagemap=on;

proc sort data=import;

by Fwd1Bk2;

title "Poisson regression with repeated measurement";

proc glimmix data=import plots=StudentPanel(conditional) nobound;

class AthleteID GameID;

model EffectiveRucks=MTPforceStdzd WeekRescaled

/link=log dist=poisson offset=LnTimePlayed;

estimate "Mean" int 1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ Week 1" int 1 WeekRescaled -0.5/cl exp alpha=0.1;

estimate "Mean @ Week 6" int 1 WeekRescaled 0.5/cl exp alpha=0.1;

estimate "Week 6/Week 1" WeekRescaled 1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ -1SD MTPf" int 1 MTPforceStdzd -0.5/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 MTPforceStdzd 0.5/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" MTPforceStdzd 1/cl exp alpha=0.1;

random int/subject=AthleteID cl alpha=0.1;

random GameID;

random \_residual\_;

covtest/wald cl(alpha=0.1);

by Fwd1Bk2;

run;

The **proc sort** is in preparation for doing the analysis **by Fwd1Bk2**.

The **model** statement is straightforward, addressing the question of the extent to which MTPforce affects the count of effective rucks, and the extent to which there is a linear trend in effective rucks over the six weekly games. **link=log dist=poisson** are required for a dependent variable that is a count, and the **offset** causes the analysis to estimate counts per 80-minute game. (If a player played only 40 min and got 4 effective rucks, his count would be effectively 8 per game.) You have to use the natural log of TimePlayed, because the model predictors the natural log of the counts. Sorry, I know it's hard to understand!

In the first **estimate** statement, the **int** stands for intercept (you can write it out in full as **intercept**), and this is one way to output its value. The **exp** outputs back-transformed values: a count for the intercept and other means, and count ratios (factors) for effects.

The other **estimate** statements show you how the effect of the numeric predictors (WeekRescaled and MTPforceStdzd) represent differences in means. We will also need the means to evaluate the magnitudes of the predictors via standardization.

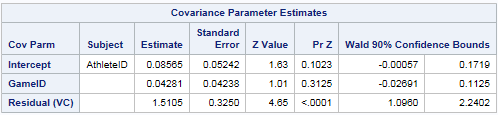
**random int/subject=AthleteID** accounts for repeated measurement on the athletes and produces a variance representing the difference between the athletes. The **cl** **alpha=0.1** produces the mean for each athlete, with 90% confidence limits. You will see that the means need to be back-transformed to factors.

**random GameID** treats the six games like a random sample of games.

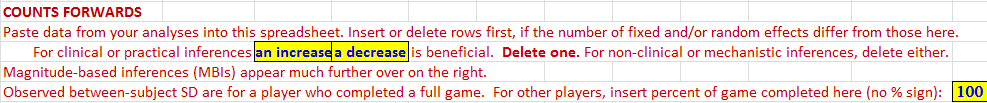
**random \_residual\_** is the way you have to tell Glimmix to allow for overdispersion in the counts.

**covtest/wald** produces confidence limits for the random-effect variances.

1. Copy the code into Program 1 and run it. The RESULTS window starts with the forwards, and Covariance Parameter Estimates (random effects) appear first:

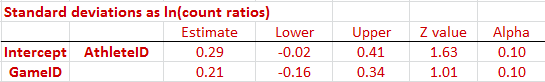


Residual (VC) is the Poisson variance overdispersion factor. The other covparms need to be back-transformed. Right-click and download **process Poisson and logistic repeated meaures.xlsx**, where you will first see this:

The current analysis requires only non-clinical inferences, so it doesn't matter whether you delete an increase or a decrease. Let's delete a decrease.

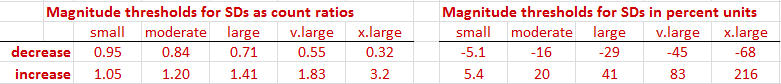
The analysis involves the observed between-player SD, which depends on the observed count, and that depends on the proportion of full time the players are on the field. The default shown of 100 (%) is for players who played a full game. You can work out the percent for the average Forward from the raw data, or use the value from the analysis of proportions later on; they differ a little, because the mean of raw data will never agree exactly with the predicted mean of log- or logistically-transformed data adjusted for whatever. The percent proportion of full (80-min) game time for the raw data is 71 for the forwards, but let's leave it at 100.

Next you will see these…



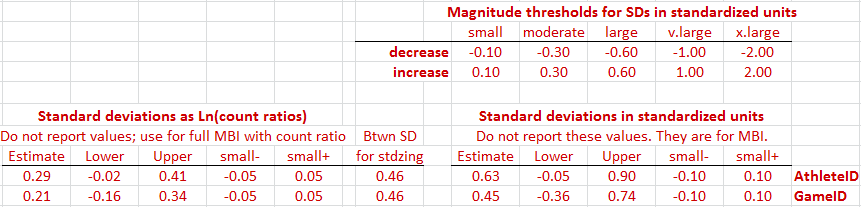
…which are just square roots of the variances. Check out the formulae in the cells, which allow for negative variance. Further over find these:  
 

Again, check the formulae. There are no occurrences of 100 in the back-transformation, because the log transformation in Glimmix is simply log, not 100\*log. Also, I've back-transformed them to *factor* SDs; that is 1.34 means athletes differ from each other typically by ×/÷1.34. The SDs are also shown in percent units: a factor of ×1.34 is 34%, but be aware that ÷1.34 is not -34%; it's actually 100/1.34-100 = -25% (not shown). How big are such differences? The full set of thresholds is shown above these two tables…



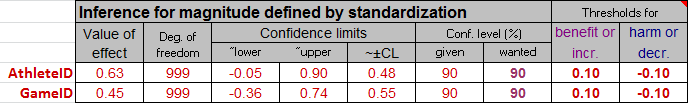
Check the formulae and you will see that the values are the square roots of the usual factor thresholds. Why? Because we're dealing here with SDs, which have half the usual thresholds in log units, which corresponds to the square root of the usual factor thresholds.

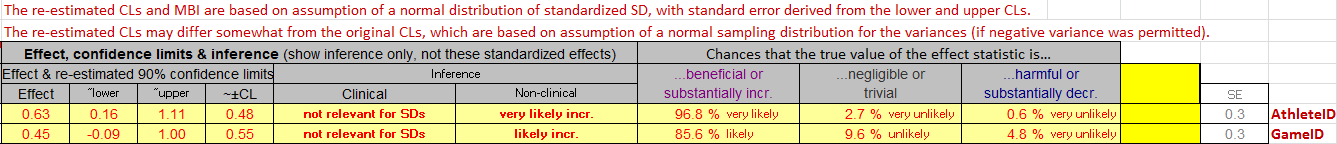
1. However, it would be wrong to use factor thresholds to assess these SD. The subjects are team-sport athletes, so we should standardize with the appropriate observed between-subject SD. The calculations and thresholds are next on the right:



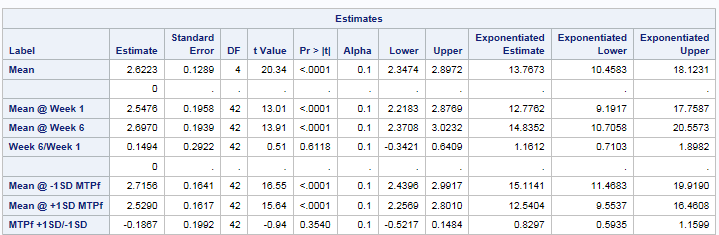
The Ln(count ratios) are just copied across from the far left of the spreadsheet. Much more difficult is the Btwn SD for stdzing. This is the square root of the sum of the pure between-subject variance (in log units) and the sampling variance of the log of an over-dispersed Poisson distribution, which is given by the overdispersion factor times the inverse of the appropriate count (which can be demonstrated by statistical first principles). I have made the appropriate count equal to half a pure between-subject SD below the overall mean count (the overall mean times the square root of the pure SD in factor units), because an SD represents a difference between subjects, so it seems only reasonable to "center" the two subjects on the overall mean. I am sorry this turned out to be so complicated. It's even worse when we get to standardizing effects for proportion ratios! Anyway, the rest is easy: the Standard deviations as Ln(count ratios) are divided by the Btwn SD for stdzing to give the standard deviations in standardized units shown above, and the magnitude thresholds are simply half the usual values. Click on the cells to check.

You can see that the pure between-athlete SD (the value for AthleteID) is large and clear, whereas the SD for the differences between games (GameID) is moderate and unclear. The full MBIs are shown further over on the right, here split into two:

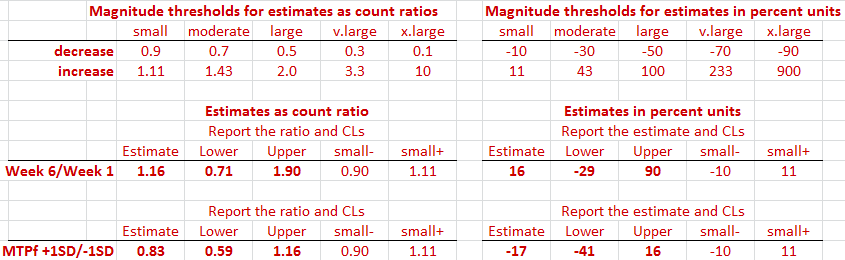




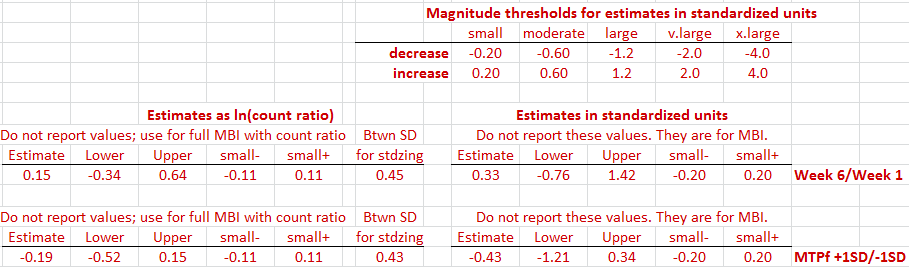
Look carefully and you will notice the re-estimated confidence limit (in the pale yellow cells) have changed, although the ± form of the confidence limits is the same. The explanation is give in red above the cells: both sets are approximations arising from assumptions about the sampling distributions of the variance and the standard deviation. Don't lose sleep over this issue. Go with the inferences shown in pale yellow cells. Note also that clinical inferences are not appropriate for SDs.

1. In the OUTPUT window, scroll down to the Estimates:  
   

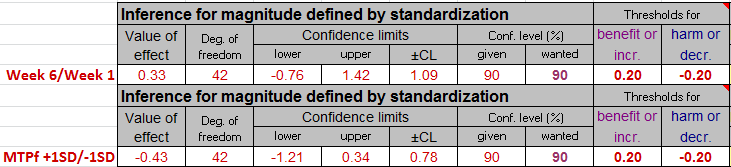
Let's see if there is a substantial trend over the six weeks and a substantial effect of the strength test. The exponentiated (i.e., back-transformed) estimates are what we want. The value for Mean is the mean count of effective rucks in an 80-min game, when WeekRescaled=0 (i.e., the middle of the season of six games) and MTPforceStdzd=0 (i.e., for the athlete with average thigh pull force). Mean @ Week 1 and Mean @ Week 6 are for athletes with average thigh pull force. Mean @ -1SD MTPf and Mean @ -1SD MTPf are for the middle of the season. The other two exponentiated estimates are factors. I've copied the table into the spreadsheet and reproduced the factors along with their percent units and magnitude thresholds:

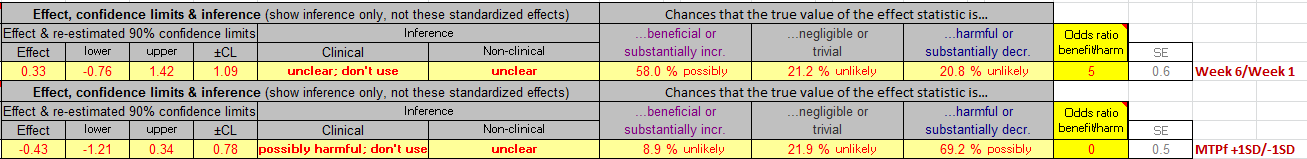


The Week effect represents a factor increase of 1.16 over the six games, while MTPforceStdzd represents a factor decrease of 0.83. These would not be clear if factor thresholds were appropriate, but we have to use standardization. So over on the right find this…



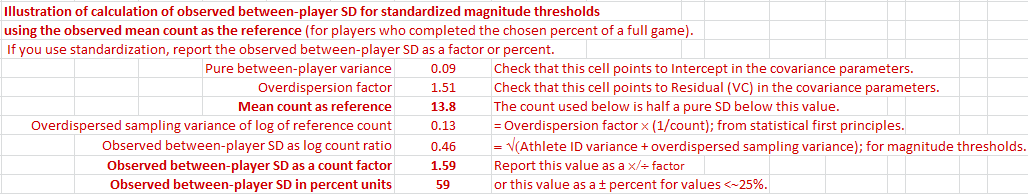
…which shows both effects are small but unclear. Click on the Btwn SD for stdzing to see that it comes from the sum of the pure between-subject variance and the over-dispersed Poisson sampling variance estimated for the reference count. The full MBI confirms that the effects are unclear:





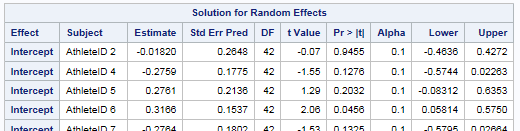
Obviously more games or subjects are needed to improve precision. Note that there is no inconsistency with the original and re-estimated confidence limits for fixed effects.

1. Scroll back to the left and down a bit to find a detailed explanation of the derivation of the observed between SD:

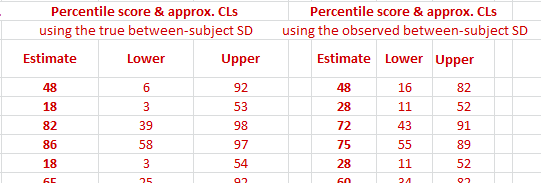


Study these cells carefully for a deeper understanding.

1. Finally, the Solution for Random Effects:



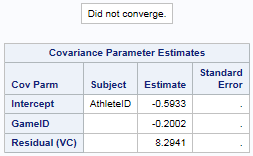
These are shown in the spreadsheet, with the estimates and confidence limits converted to percentile scores using the true and observed between-subject SDs:



This is a cool way to properly assess individual athletes! I don't know which is more appropriate, but I am leaning towards using the observed SD.

Beware, these assessments are AFTER adjustment for the strength test. If the fitness test explains some of the performance indicator, then the above assessments do not reflect the athletes' ability on the field. You need to run the analyses with the fitness test removed from the model.

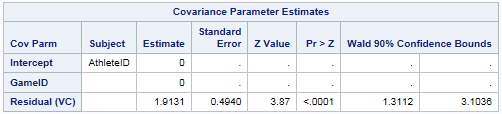
1. Moving on to the backs, you'll see a disaster:



I sometimes get around this problem by relaxing the convergence criterion, which is specified with pconv in the proc glimmix statement. The default is pconv=1e-8 (i.e., 10-8). I tried pconv=1e-6 and pconv=1e-4 without success. Obviously 10 backs and six games don't represent enough data to allow estimation of negative variance.

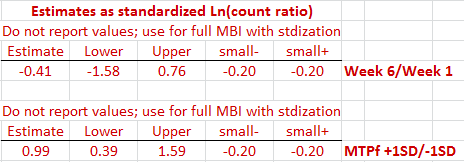
Next thing to try is allowing only positive variance. So remove the nobound from the proc glimmix statement, run the analysis again, and scroll down to the backs.

1. Success! However, you get this enigmatic warning:   
     
   which is what you see when one or more of the random effects is zero:

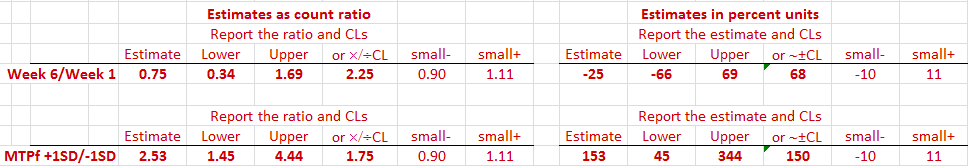


The zeros mean there are no individual values for the backs or the games, as shown further down for the random-effect solution for the players. You can conclude that, with this sample, the variability players showed from game to game was consistent with no real differences between players or games.

1. I've processed the output for the backs by copying the forwards' spreadsheet (press Ctrl and click-drag the tab) and overwriting the forwards' data with the backs' data. The estimates show a much lower mean count of effective rucks than we saw for the forwards. Here are the standardized effects:



The trend over the six games is a moderate but unclear reduction, while the effect of thigh pull force is clear and moderate. Once again, you don't show these values in a manuscript, but you do show either the factor (ratio) or percent effects:



Ignore the smallest effects in these tables.

Evidently, this effect of the fitness test has explained all of the differences between the players (the zero variance for players). What we should do at this point is rerun the analysis with MTPforceStdzd removed from the model, so we see what kind of differences there are between players before the fitness test is taken into account. I turned on Track changes in Word so you can see the changes in the program. Copy and paste the following code into SAS Studio below the previous program, and run it:

title "Poisson regression with repeated measurement";

title2 "MTPforceStdzd removed from the model";

proc glimmix data=import plots=StudentPanel(conditional);

class AthleteID GameID;

model EffectiveRucks=WeekRescaled

/link=log dist=poisson offset=LnTimePlayed;

estimate "Mean" int 1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ Week 1" int 1 WeekRescaled -0.5/cl exp alpha=0.1;

estimate "Mean @ Week 6" int 1 WeekRescaled 0.5/cl exp alpha=0.1;

estimate "Week 6/Week 1" WeekRescaled 1/cl exp alpha=0.1;

/\*

estimate "";

estimate "Mean @ -1SD MTPf" int 1 MTPforceStdzd -0.5/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 MTPforceStdzd 0.5/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" MTPforceStdzd 1/cl exp alpha=0.1;

\*/

random int/subject=AthleteID cl alpha=0.1;

random GameID;

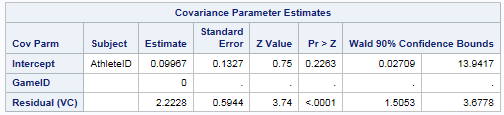
random \_residual\_;

covtest/wald cl(alpha=0.1);

by Fwd1Bk2;

run;

1. Scroll down to the backs and to see substantial variance for the players but no still between-game variance:



I haven't copied the output into a spreadsheet. You could easily overwrite the previous values for the backs.

1. Let's now address the question of comparing the means of forwards and backs, and comparing the values of their covariates. You could do that with the [Combine/compare effects](http://www.sportsci.org/resource/stats/xCombineGroups.xls) spreadsheet at Sportscience, but strictly speaking it would not be correct to do so, because the forwards and backs are not completely independent of each other: they went to the same games. To properly account for this level of repeated measurement, you need to put forwards and backs into the same analysis. I tried it initially with nobound and with random \_residual\_/group=Fwd1Bk2; to give different overdispersion for the two groups, but I got the dreaded Did not converge. The slightly simpler code that worked is shown below, with the additions from the previous programs highlighted.

Note that I have compared the means of the forwards and backs with an lsmeans statement, which includes ilink (inverse link) to back-transform the estimates to means. I have also specified the same means and their difference with estimate statements: I did it this way to get the mean for the backs, which is required to get their between-subject SD for the comparison of the means via standardization. The lsmeans is therefore redundant, but it's good to check that the estimate statements are correct. I have also included it to show you the syntax for any future analyses.

title "Poisson regression with repeated measurement";

title2 "with Fwd2Bk2 in a reasonably complete model";

proc glimmix data=import plots=StudentPanel(conditional);

class AthleteID GameID Fwd1Bk2;

model EffectiveRucks=Fwd1Bk2 Fwd1Bk2\*MTPforceStdzd WeekRescaled

/link=log dist=poisson offset=LnTimePlayed;

lsmeans Fwd1Bk2/diff cl alpha=0.1 ilink;

estimate "Mean Fwds&Backs" int 1 Fwd1Bk2 0.5 0.5/cl exp alpha=0.1;

estimate "";

estimate "Mean Backs" int 1 Fwd1Bk2 0 1/cl exp alpha=0.1;

estimate "Mean Fwds" int 1 Fwd1Bk2 1 0/cl exp alpha=0.1;

estimate "Mean Fwds/Backs" Fwd1Bk2 1 -1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ Week 1" int 1 WeekRescaled -0.5/cl exp alpha=0.1;

estimate "Mean @ Week 6" int 1 WeekRescaled 0.5/cl exp alpha=0.1;

estimate "Week 6/Week 1" WeekRescaled 1/cl exp alpha=0.1;

estimate "Backs:";

estimate "Mean @ -1SD MTPf" int 1 Fwd1Bk2 0 1 Fwd1Bk2\*MTPforceStdzd 0 -0.5/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 Fwd1Bk2 0 1 Fwd1Bk2\*MTPforceStdzd 0 0.5/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" Fwd1Bk2\*MTPforceStdzd 0 1/cl exp alpha=0.1;

estimate "Fwds:";

estimate "Mean @ -1SD MTPf" int 1 Fwd1Bk2 1 0 Fwd1Bk2\*MTPforceStdzd -0.5 0/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 Fwd1Bk2 1 0 Fwd1Bk2\*MTPforceStdzd 0.5 0/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" Fwd1Bk2\*MTPforceStdzd 1 0/cl exp alpha=0.1;

estimate "Backs/Fwds:";

estimate "Mean Backs+Fwds reference" int 1 Fwd1Bk2 0.5 0.5

Fwd1Bk2\*MTPforceStdzd -0.5 -0.5/cl exp alpha=0.1;

estimate "blank";

estimate "MTPf +1SD/-1SD" Fwd1Bk2\*MTPforceStdzd -1 1/cl exp alpha=0.1;

random int/subject=AthleteID cl alpha=0.1;

random GameID;

random \_residual\_;

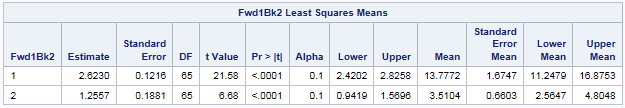
covtest/wald cl(alpha=0.1);

\*by Fwd1Bk2;

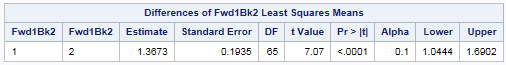
run;

The estimate statements for the effects of MTPforceStdzd are quite complicated. Comparison of the effect of MTPforceStdzd between backs and forwards via standardization is also tricky: we should probably compare the standardized effect for the forwards with the standardized effect for the backs; however, that is just too difficult to achieve, because we would have to specify two mean values to derive an appropriate between-subject SD. I have opted instead to use the mean of the backs and forwards at -1SD of MTPforce to generate the appropriate between-subject SD.

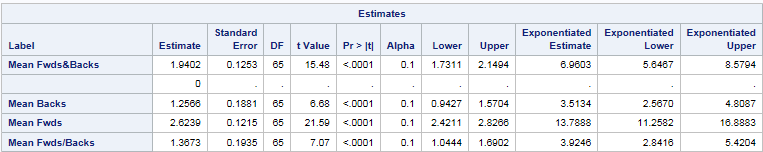
1. Try to understand all this new code, then run the program and check out the random effects (not shown here). Here are the least-squares means…



…and the difference between the least-squares means…

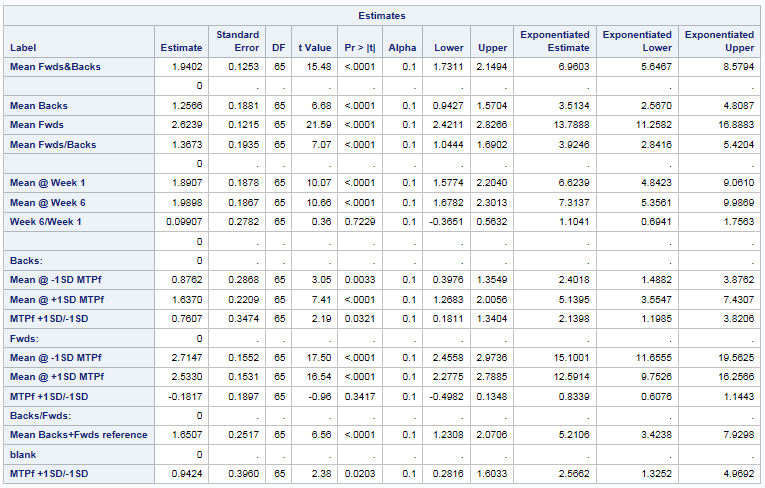


…which are exactly the same as the values in the Estimates…

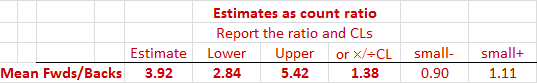
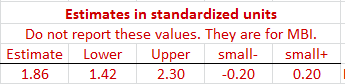


… so I have not put the least-squares or their difference into the spreadsheet.

1. Here is the full set of estimates…



The spreadsheet shows that the comparison of the means for forwards and backs via standardization is clear and large: the forwards did 3.9 times as many effective rucks as the backs, and the effect is large and clear:

The comparison of the effect of MTPforce is also large and clear. It was hard to understand as Fwds/Backs, which is why I wrote it in the program as Backs/Fwds:

(Ignore the ratio thresholds of 0.90 and 1.11 in the above two comparisons.) Get the full MBI from the cells for standardized effects over on the right.

# Logistic Regression with Repeated Measurement

1. Now let's modify the model to investigate the linear relationship between the fitness test and the *proportion* of effective rucks. Here is the code, with the changes from the first program tracked:

title "Logistic regression with repeated measurement";

proc glimmix data=import plots=StudentPanel(conditional) nobound;

class AthleteID GameID;

model EffectiveRucks/TotalRucks=MTPforceStdzd WeekRescaled

/link=logit dist=binomial;

estimate "Mean" int 1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ Week 1" int 1 WeekRescaled -0.5/cl exp alpha=0.1;

estimate "Mean @ Week 6" int 1 WeekRescaled 0.5/cl exp alpha=0.1;

estimate "Week 6/Week 1" WeekRescaled 1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ -1SD MTPf" int 1 MTPforceStdzd -0.5/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 MTPforceStdzd 0.5/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" MTPforceStdzd 1/cl exp alpha=0.1;

random int/subject=AthleteID cl alpha=0.1;

random GameID;

random \_residual\_;

covtest/wald cl(alpha=0.1);

by Fwd1Bk2;

run;

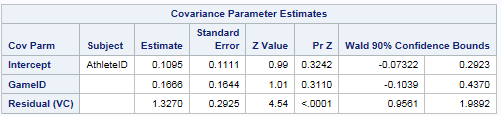
This time nobound worked for forwards and backs.

The model statement shows one of the two ways of stating a proportion as the dependent, the so-called *events/trials* syntax. You can also have a binary variable as the dependent.

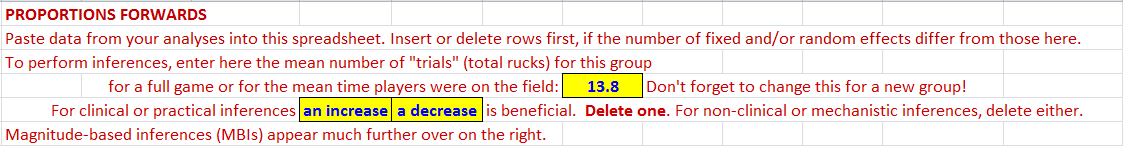
logit stands for logistic, indicating that the model is predicting the log of the odds of the dependent, and binomial is the distribution of a proportion.

Copy the code into Program 1 and run it.

1. Here are the covparms for the forwards:



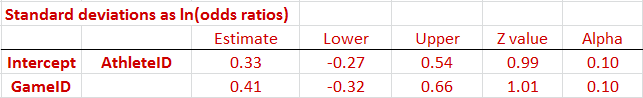
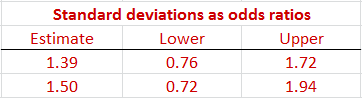
When you click on the Proportions Forwards tab of **process Poisson and logistic repeated meaures.xlsx** to see what to with these, you will find this:



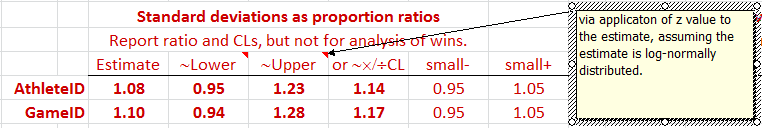
Once again the inferences are non-clinical, so you can delete an increase or a decrease. Delete a decrease.

You also have to include the mean number of trials, the denominator in the proportion represented by the events/trials dependent variable. This number is used to derive the sampling variation of the proportion, which is needed for standardizing.

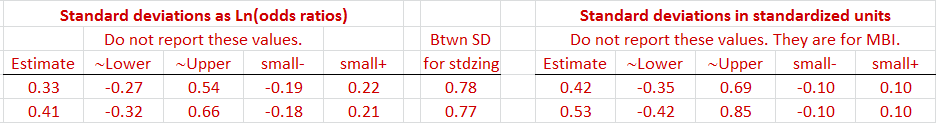
Now scroll down to see that, after square rooting and back-transformation, the covparms are standard deviations expressed as odds ratios:

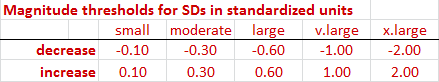
For evaluation of magnitudes, they have to be converted to proportion ratios; that is, the ratio of the proportions of athletes who are separated by 1 SD, and that has to be done at the mean proportion by calculating the proportions above and below the mean by 0.5 SD (actually it's above and below by a factor of √SD, all done via odds). The confidence limits are derived by applying the z score for the variance to the estimate of the proportion ratio, assuming that the proportion ratio is log-normally distributed:



1. Don't evaluate the magnitude with the thresholds shown here; these are for analyses of proportions of injuries or other measures where the magnitudes are defined by proportion ratios. Here the proportion ratios are for team-sport athlete, so they have to be evaluated via standardization of the log of the odds ratios. Here are the values of the Ln(odds ratios), the between-subject SD used for standardizing, and the resulting standardized values, with the thresholds for smallest effects (half of the usual 0.20, of course, because we are evaluating SDs):

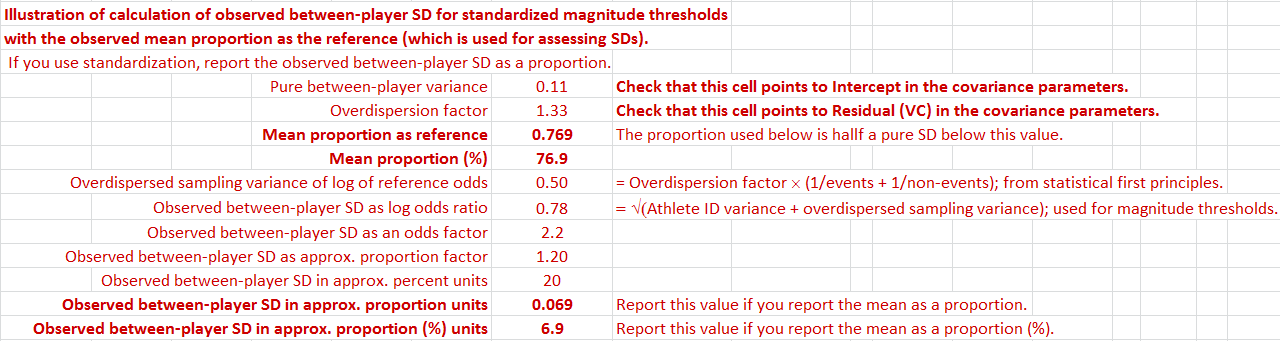
 

And when you compare the values with the full set of thresholds…

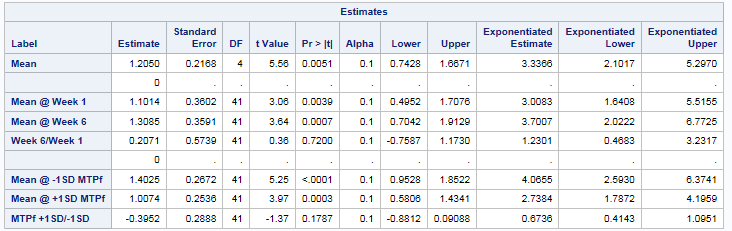


…you can see that both SDs are moderate but unclear!

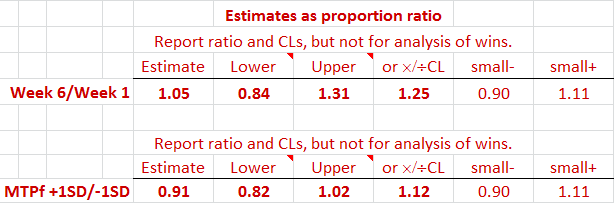
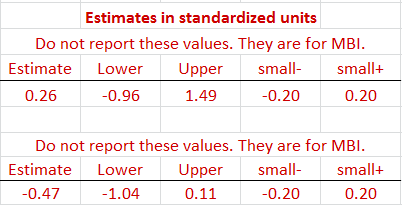
1. The derivation of the between-subject SD for standardizing is similar to that for counts. A full explanation is given below the panel for the fixed-effect estimates:



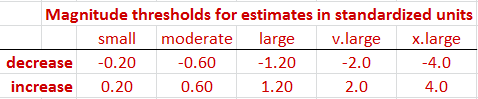
1. While you are down here, scroll down a bit further to see the solution for the AthleteID random effect. Although the SD for AthleteID was unclear, we can still assess the athletes via percentile transformations of the solution. The confidence interval for each athlete is quite wide, which is consistent with the uncertainty in the SD for AthleteID. Different athletes come out on top compared with the counts of effective rucks, and only Athlete 7 is still at the bottom.
2. Now scroll back up to the SDs. On the far right are the SDs expressed as proportion (%) differences for analyses where the dependent variable represents wins/losses in games or matches. The odds ratios can then be converted to proportion differences centered on 50%, which represents an evenly matched game or match. Do not use these columns for the present data.
3. Here are the fixed-effect Estimates for the forwards:



The exponentiated estimates for the means are odds, while those for the Week trend and thigh pull force are odds ratios. The spreadsheet converts these to proportions and proportion ratios for evaluation via ratio thresholds, standardization, and proportion differences. We are using standardization, so report the effects as proportion ratios with their CLs (but ignore the thresholds), and inspect the effects as standardized logs of the odds ratios (but do not report the values):

And when you compare with the full set of thresholds…



…you can see that the week effect is small but unclear, while thigh pull force has a clear small negative effect on the proportion of effective rucks.

By the way, the confidence limits for the proportion ratios were derived in the same manner as those for the SDs, by assuming the proportion ratio is log-normally distributed, then applying the t value for the log of the odds ratio to log of the proportion ratio.

1. I've processed the backs in the same manner. The differences between athletes are moderate but unclear. GameID has a small *negative* but unclear SD. The effects of week and thigh pull force are both trivial and unclear.
2. Finally a program that combines the backs and forwards in one analysis, showing changes from the program we used for the counts:

title "Logistic regression with repeated measurement";

title2 "with Fwd2Bk2 in a reasonably complete model";

proc glimmix data=import plots=StudentPanel(conditional);

class AthleteID GameID Fwd1Bk2;

model EffectiveRucks/TotalRucks=Fwd1Bk2 Fwd1Bk2\*MTPforceStdzd WeekRescaled

/link=logit dist=binomial;

lsmeans Fwd1Bk2/diff cl alpha=0.1 ilink;

estimate "Mean Fwds&Backs" int 1 Fwd1Bk2 0.5 0.5/cl exp alpha=0.1;

estimate "";

estimate "Mean Backs" int 1 Fwd1Bk2 0 1/cl exp alpha=0.1;

estimate "Mean Fwds" int 1 Fwd1Bk2 1 0/cl exp alpha=0.1;

estimate "Mean Fwds/Backs" Fwd1Bk2 1 -1/cl exp alpha=0.1;

estimate "";

estimate "Mean @ Week 1" int 1 WeekRescaled -0.5/cl exp alpha=0.1;

estimate "Mean @ Week 6" int 1 WeekRescaled 0.5/cl exp alpha=0.1;

estimate "Week 6/Week 1" WeekRescaled 1/cl exp alpha=0.1;

estimate "Backs:";

estimate "Mean @ -1SD MTPf" int 1 Fwd1Bk2 0 1 Fwd1Bk2\*MTPforceStdzd 0 -0.5/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 Fwd1Bk2 0 1 Fwd1Bk2\*MTPforceStdzd 0 0.5/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" Fwd1Bk2\*MTPforceStdzd 0 1/cl exp alpha=0.1;

estimate "Fwds:";

estimate "Mean @ -1SD MTPf" int 1 Fwd1Bk2 1 0 Fwd1Bk2\*MTPforceStdzd -0.5 0/cl exp alpha=0.1;

estimate "Mean @ +1SD MTPf" int 1 Fwd1Bk2 1 0 Fwd1Bk2\*MTPforceStdzd 0.5 0/cl exp alpha=0.1;

estimate "MTPf +1SD/-1SD" Fwd1Bk2\*MTPforceStdzd 1 0/cl exp alpha=0.1;

estimate "Backs/Fwds:";

estimate "Mean Backs+Fwds reference" int 1 Fwd1Bk2 0.5 0.5

Fwd1Bk2\*MTPforceStdzd -0.5 -0.5/cl exp alpha=0.1;

estimate "blank";

estimate "MTPf +1SD/-1SD" Fwd1Bk2\*MTPforceStdzd -1 1/cl exp alpha=0.1;

random int/subject=AthleteID cl alpha=0.1;

random GameID;

random \_residual\_;

covtest/wald cl(alpha=0.1);

\*by Fwd1Bk2;

run;

1. The program runs OK, and I have included a spreadsheet for processing it, which you can use as a template for your own data.
2. So is it all too difficult? What's the alternative? Just deciding whether an effect is significant or not is no longer tenable. There are no magnitude thresholds for the odds ratio, so you have to convert it into another effect statistic with such thresholds.
3. There is one simpler alternative: invoke the central-limit theorem and process counts and proportions with Proc Mixed as if they were an unproblematic continuous variable. It probably won't work well if there are proportions near or at zero or 1, and you certainly can't use log transformation if there are zeros. Someone (me, presumably) should investigate how well this simpler approach stacks up against full Poisson and logistic regression for the kinds of data we encounter in sport and exercise science.