

Simpson's Paradox:
The Trapezoid of Means
with Paik-Agresti and Liu-Meng Diagrams

MATH 4939

January 2026

Last update: January 11 2026 23:20

Contents

1	Setup	2
2	Estimating other parameters	24
2.1	Refitting with equivalent formulas	24
2.2	Using linear transformations and Wald tests	27
3	The Paik-Agresti Diagram	32
4	Exercise 1: How a superficial model can hide inequity.	42
5	Exercise 2: Simpson's Paradox and Choosing a Restaurant: A hypothetical example (tricky!)	42
6	The Liu-Meng Diagram: The main effect of Z	45

List of Figures

1	The red lines show the fitted values using a model without an interaction term	23
---	--	----

1 Setup

This R script is designed to be run either interactively one line at a time, or as a Rmarkdown script that creates a pdf file.

In a clean version of a report, this portion of the Rmarkdown script would be hidden by using a ‘include=FALSE’ option for the ‘chunk header’. That is, the header line for the chunk would read:

```
#+ setup, include=FALSE
```

```
knitr::opts_chunk$set(comment = ' ',fig.height = 3.2,fig.width=5)
```

```
#  
# For interactive use, the following sets the  
# working directory to this file's location.  
#  
setwd(this.path::here())  
#  
library(lattice)  
library(latticeExtra)  
library(spida2)    # remotes::install_github('gmonette/spida2')
```

This is made-up data designed to be consistent with a study reported on CBC radio some years ago.

A physician was interviewed who reported concerns over the use of sunscreen lotion because it was found that users of sunscreen lotion had a higher incidence of skin cancer than non-users. It was suggested that sun screen lotion might contain some harmful chemicals.

The following simulated data is intended to help explore possible explanations. Do your own research to find out whether there is a current consensus on risk factors associated with the use of sunscreen lotion.

```
# Direct text input of a data frame:
ddin <- read.table(header = TRUE, text = "
SkinDamage.mean  SunScreen SunExposure    n
        6         1         1    50
        9         0         1    10
        3         1         0    10
        4         0         0    70
")
dim(ddin)
```

```
[1] 4 4
```

```
# Creating a big data frame by repeating rows of a small one:
dd <- ddin[rep(1:nrow(ddin), ddin$n), ] # What does this do?
dim(dd)
```

```
[1] 140  4
```

```
# Generate a random random seed
sample(10000000,1)
```

```
[1] 6415749
```

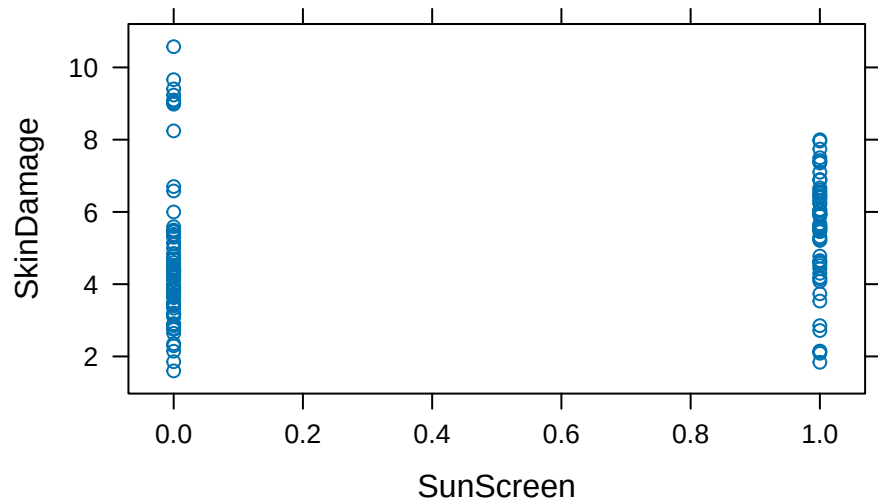
```
set.seed(9607680) # and use the same one for reproducibility
dd$eps <- rnorm(nrow(dd)) # generate random epsilons

dd$SkinDamage <-
  with(dd, SkinDamage.mean +
    ave(eps, SunExposure, SunScreen, FUN = scale)) # What does this do?
```

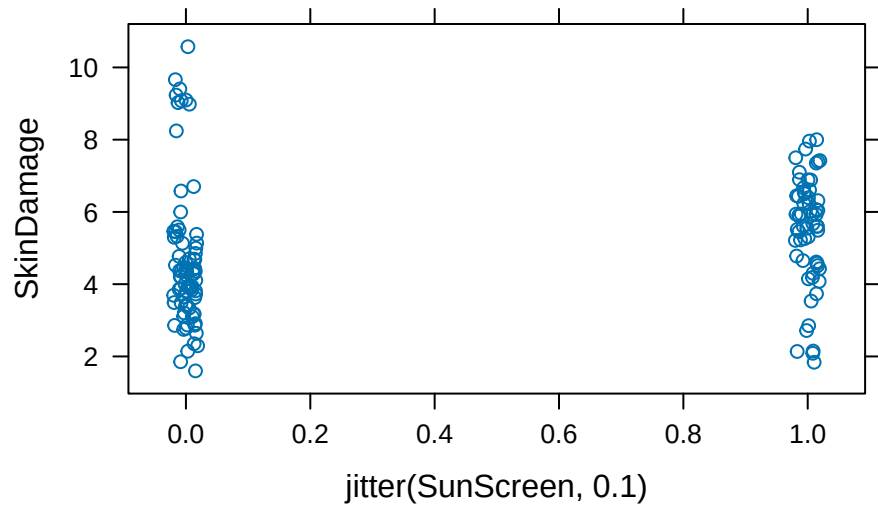
The function ‘ave’ is a very powerful tool that applies the argument of FUN to each chunk of `eps` within each combination of values of `SunExposure` and `SunScreen`. The function `scale` rescales its input (with a location-scale transformation) so the result has mean 0 and standard deviation 1. Thus, the **sample average** of `SkinDamage` is equal to `SkinDamage.mean`. Otherwise, the sample average would differ by a small random amount.

The only purpose in doing this here is to make the results come out as nice round numbers for presentation in class.

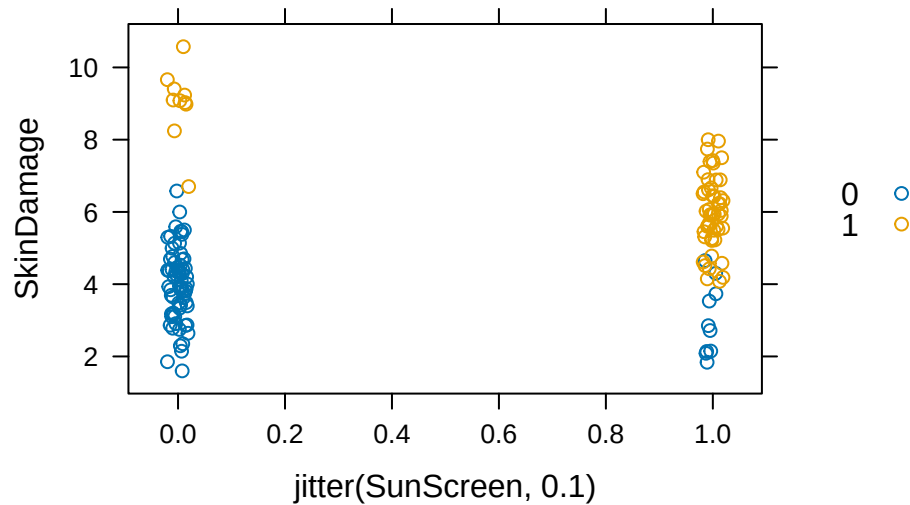
```
xyplot(SkinDamage ~ SunScreen, dd)
```



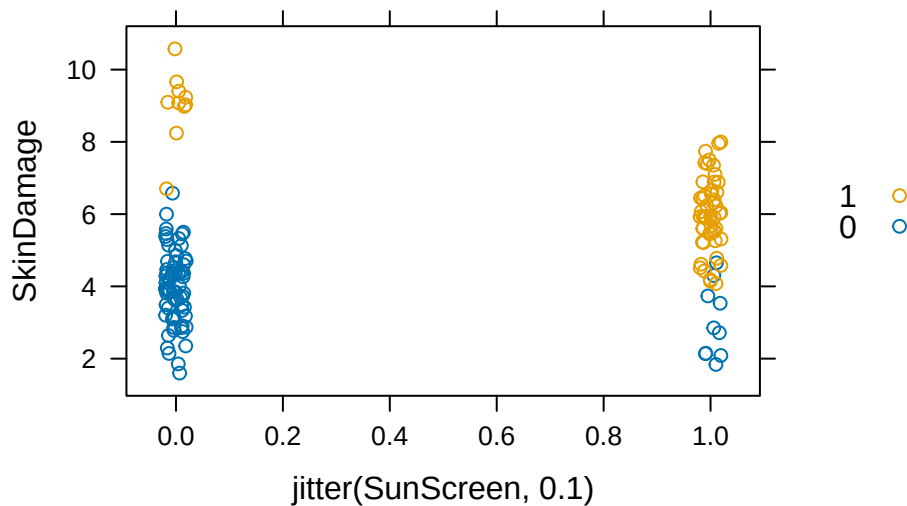
```
xyplot(SkinDamage ~ jitter(SunScreen, .1), dd)
```



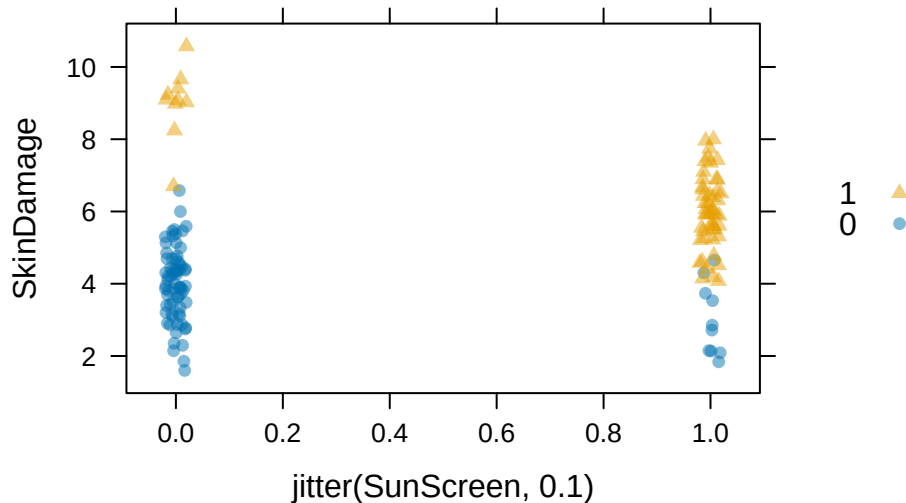
```
xyplot(SkinDamage ~ jitter(SunScreen, .1), dd, groups = SunExposure, auto.key = T)
```



```
xyplot(SkinDamage ~ jitter(SunScreen, .1), dd,  
       groups = SunExposure, auto.key = list(reverse.rows = T)) # why do this?
```

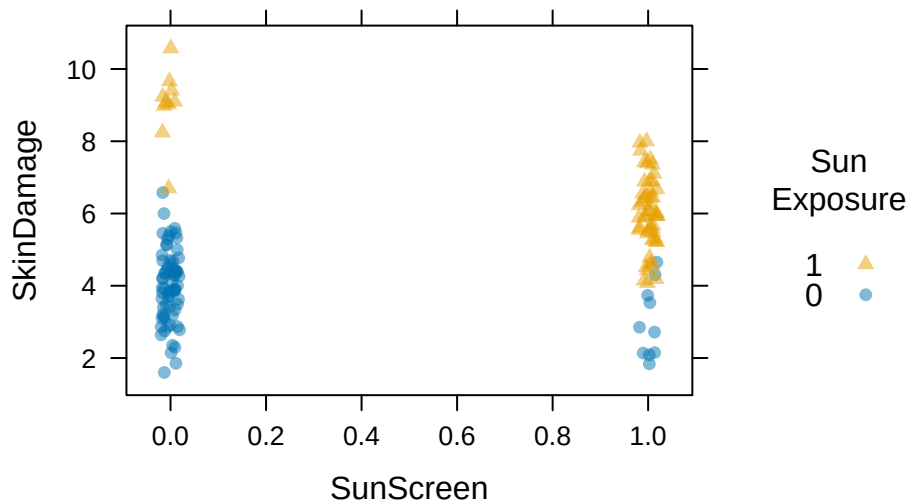


```
xyplot(SkinDamage ~ jitter(SunScreen, .1), dd,
       groups = SunExposure, auto.key = list(reverse.rows = T),
       par.settings =
         list(superpose.symbol = list(pch = c(16,17),alpha = .5))) # why do this?
```



```
p <- xyplot(SkinDamage ~ jitter(SunScreen, .1), dd,  
            groups = SunExposure,  
            auto.key =  
              list(reverse.rows = T, title = 'Sun\nExposure', cex.title = 1),  
            par.settings = list(superpose.symbol = list(pch = c(16,17), alpha = .5)),  
            xlab = 'SunScreen')
```

p



How helpful is SunScreen?

```
fit <- lm(SkinDamage ~ SunScreen, dd)
summary(fit)
```

Call:

```
lm(formula = SkinDamage ~ SunScreen, data = dd)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-3.6608	-1.0162	-0.2041	0.7693	5.9484

Coefficients:

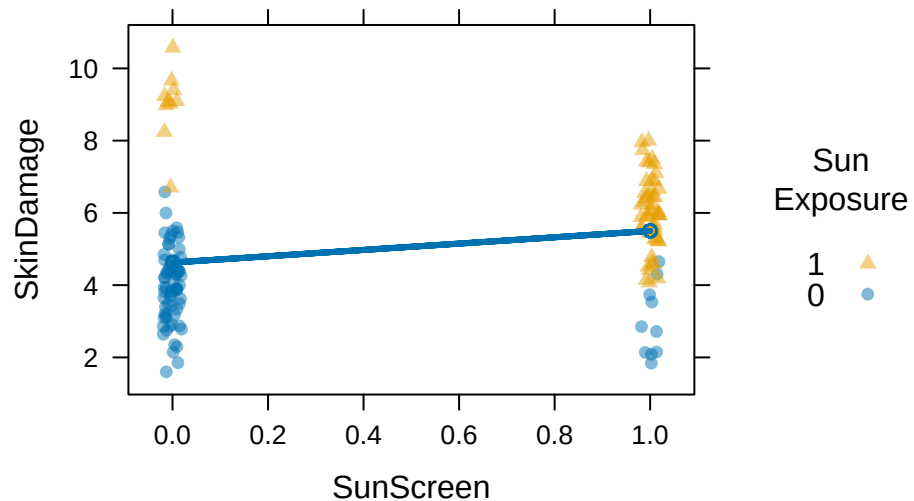
	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	4.6250	0.1973	23.442	<2e-16	***
SunScreen	0.8750	0.3014	2.903	0.0043	**

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.765 on 138 degrees of freedom
Multiple R-squared: 0.05757, Adjusted R-squared: 0.05074
F-statistic: 8.429 on 1 and 138 DF, p-value: 0.0043

```
dd$fit <- predict(fit)
```

```
p + xyplot(fit ~ SunScreen, dd, type = 'b', lwd = 3)
```



It looks like using SunScreen results in higher levels of Skin Damage What if we 'control' for SunExposure

```
fit2 <- lm(SkinDamage ~ SunScreen * SunExposure, dd)
summary(fit2)           # What does this mean??
```

Call:

```
lm(formula = SkinDamage ~ SunScreen * SunExposure, data = dd)
```

Residuals:

Min	1Q	Median	3Q	Max
-2.39988	-0.67100	-0.00626	0.60225	2.57896

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	4.0000	0.1195	33.466	< 2e-16 ***
SunScreen	-1.0000	0.3381	-2.958	0.00365 **
SunExposure	5.0000	0.3381	14.790	< 2e-16 ***

```
SunScreen:SunExposure  -2.0000      0.4840  -4.132 6.25e-05 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1 on 136 degrees of freedom
```

```
Multiple R-squared:  0.7018,    Adjusted R-squared:  0.6952
```

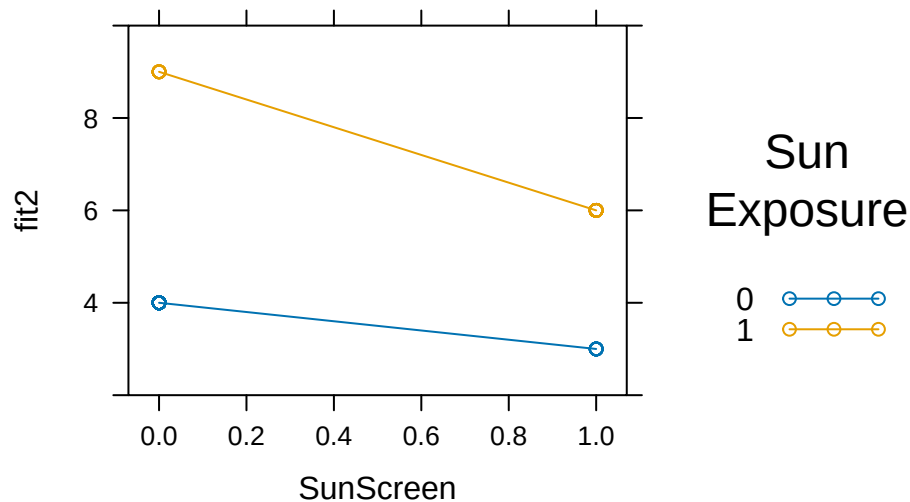
```
F-statistic: 106.7 on 3 and 136 DF,  p-value: < 2.2e-16
```

Interpreting output: Just think of partial derivatives!

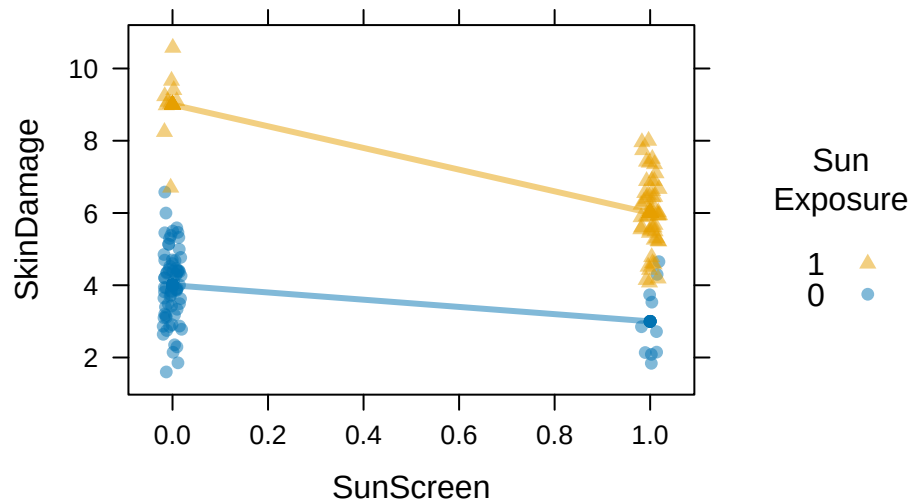
```
dd$fit2 <- predict(fit2)
spida2::up(subset(dd, select = c(fit2, SunExposure, SunScreen)),
  ~ SunExposure + SunScreen)
```

	fit2	SunExposure	SunScreen	Freq
0/0	4	0	0	70
0/1	3	0	1	10
1/0	9	1	0	10
1/1	6	1	1	50

```
xyplot(fit2 ~ SunScreen, dd, type = 'b', groups = SunExposure,  
       ylim = c(2,10), auto.key = list(title = 'Sun\nExposure'))
```



```
p + xyplot(fit2 ~ SunScreen, dd, type = 'b', groups = SunExposure, lwd = 3)
```



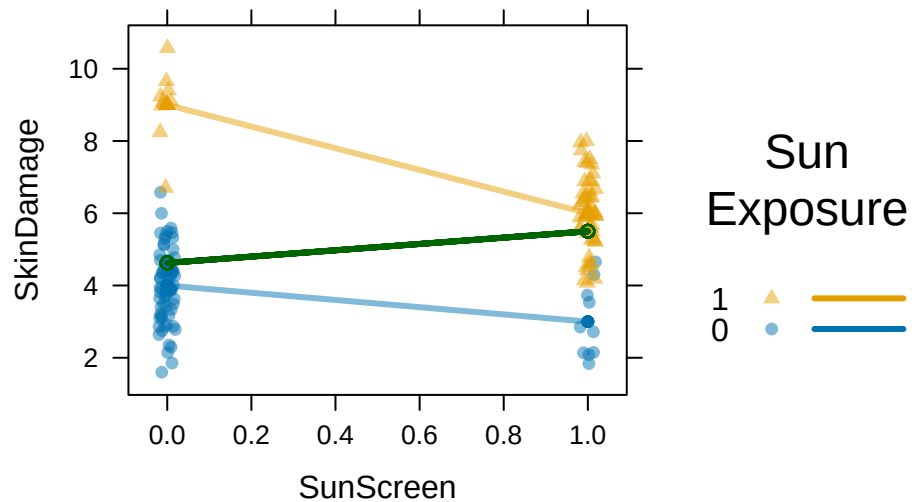
```
pall <- update(                                # creates the plot  
  p, legend = NULL,
```

```

auto.key= list(lines = T, title = 'Sun\nExposure', reverse.rows = TRUE),
par.settings = list(
  plot.line = list(lwd = 3),
  superpose.line = list(lwd = 3)
)
) +
xyplot(fit2 ~ SunScreen, dd, type = 'b', groups = SunExposure) +
xyplot(fit ~ SunScreen, dd, type = 'b', col = 'darkgreen')

```

```
pull # shows the plot
```



What does a model look like without interactions?

```
fitni <- lm(SkinDamage ~ SunScreen + SunExposure, dd)
dd$fitni <- predict(fitni)
```

```
update(p, legend = NULL, auto.key= list(title = 'Sun Exposure')) +
  xyplot(fit2 ~ SunScreen, dd, type = 'b', groups = SunExposure) +
  xyplot(fit ~ SunScreen, dd, type = 'b', col = 'darkgreen') +
  xyplot(fitni ~ SunScreen, dd, groups = SunExposure, type = 'b',
        col = 'red')
```

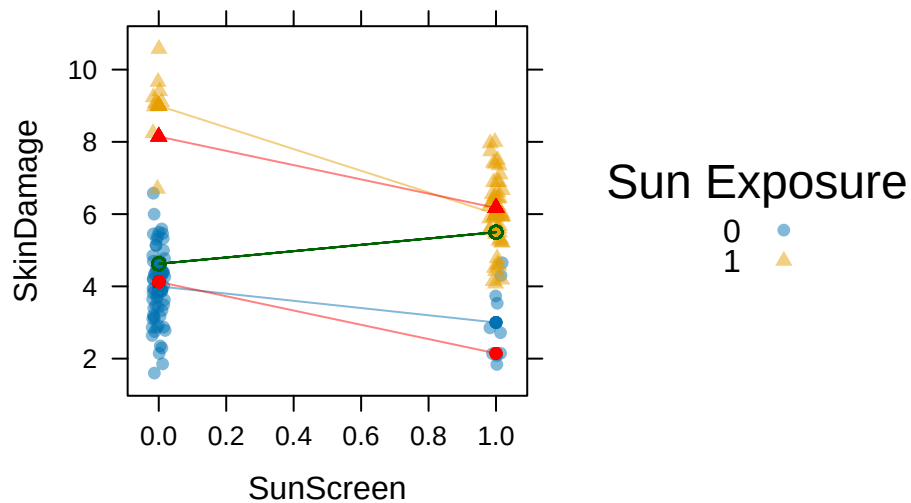


Figure 1: The red lines show the fitted values using a model without an interaction term

2 Estimating other parameters

Two approaches:

- Refit with a different but equivalent formula
- Use linear transformations and a Wald test

2.1 Refitting with equivalent formulas

```
fit_conditional1 <-  
  lm(SkinDamage ~ factor(SunExposure)/SunScreen - 1, dd)  
summary(fit_conditional1)
```

Call:

```
lm(formula = SkinDamage ~ factor(SunExposure)/SunScreen - 1,  
    data = dd)
```

Residuals:

Min	1Q	Median	3Q	Max
-----	----	--------	----	-----

-2.39988 -0.67100 -0.00626 0.60225 2.57896

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
factor(SunExposure)0	4.0000	0.1195	33.466	< 2e-16 ***
factor(SunExposure)1	9.0000	0.3162	28.460	< 2e-16 ***
factor(SunExposure)0:SunScreen	-1.0000	0.3381	-2.958	0.00365 **
factor(SunExposure)1:SunScreen	-3.0000	0.3464	-8.660	1.21e-14 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 136 degrees of freedom

Multiple R-squared: 0.9656, Adjusted R-squared: 0.9646

F-statistic: 955 on 4 and 136 DF, p-value: < 2.2e-16

```
fit_conditional2 <-
```

```
  lm(SkinDamage ~ factor(SunScreen)/SunExposure - 1, dd)  
summary(fit_conditional2)
```

Call:

```
lm(formula = SkinDamage ~ factor(SunScreen)/SunExposure - 1,  
    data = dd)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-2.39988	-0.67100	-0.00626	0.60225	2.57896

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
factor(SunScreen)0	4.0000	0.1195	33.466	< 2e-16 ***
factor(SunScreen)1	3.0000	0.3162	9.487	< 2e-16 ***
factor(SunScreen)0:SunExposure	5.0000	0.3381	14.790	< 2e-16 ***
factor(SunScreen)1:SunExposure	3.0000	0.3464	8.660	1.21e-14 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 136 degrees of freedom

Multiple R-squared: 0.9656, Adjusted R-squared: 0.9646
F-statistic: 955 on 4 and 136 DF, p-value: < 2.2e-16

2.2 Using linear transformations and Wald tests

Two functions for Wald tests

- `car::lht`
 - Advantage: can test non-zero hypotheses
 - Disadvantage: Row of hypothesis matrix must be linearly independent
- `spida2::wald` (and `walddf`)
 - Advantage:
 - * can test linearly dependent rows
 - * `spida2::walddf` creates a data frame that is easy to plot
 - * can simultaneously test subsets of coefficient matched with a regular expression
 - Disadvantage: Hypotheses must be $= 0$

```
fit <- lm(SkinDamage ~ SunScreen * SunExposure, dd)
```

```
summary(fit)
```

Call:

```
lm(formula = SkinDamage ~ SunScreen * SunExposure, data = dd)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-2.39988	-0.67100	-0.00626	0.60225	2.57896

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	4.0000	0.1195	33.466	< 2e-16 ***
SunScreen	-1.0000	0.3381	-2.958	0.00365 **
SunExposure	5.0000	0.3381	14.790	< 2e-16 ***
SunScreen:SunExposure	-2.0000	0.4840	-4.132	6.25e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 136 degrees of freedom

Multiple R-squared: 0.7018, Adjusted R-squared: 0.6952

F-statistic: 106.7 on 3 and 136 DF, p-value: < 2.2e-16

```
L <- rbind(  
  "Effect of X | Z = 0" = c(0, 1, 0, 0),  
  "Effect of X | Z = 1" = c(0, 1, 0, 1),  
  "Effect of Z | X = 0" = c(0, 0, 1, 0),  
  "Effect of X | Z = 0" = c(0, 0, 1, 1),  
  "E(Y | X = 1, Z = 1)" = c(1, 1, 1, 1) # you can keep going  
)  
  
library(spida2)  
wald(fit, L) # overall test is an F-test for all hypotheses = 0
```

	numDF	denDF	F-value	p-value			
1	4	136	955	<.00001			
			Estimate	Std.Error	DF	t-value	p-value Lower 0.95
Effect of X Z = 0	-1		0.338062	136	-2.958040	0.00365	-1.668538
Effect of X Z = 1	-3		0.346410	136	-8.660254	<.00001	-3.685047
Effect of Z X = 0	5		0.338062	136	14.790199	<.00001	4.331462

Effect of X Z = 0	3	0.346410	136	8.660254	<.00001	2.314953
E(Y X = 1, Z = 1)	6	0.141421	136	42.426407	<.00001	5.720331
Upper 0.95						
Effect of X Z = 0	-0.331462					
Effect of X Z = 1	-2.314953					
Effect of Z X = 0	5.668538					
Effect of X Z = 0	3.685047					
E(Y X = 1, Z = 1)	6.279669					

```
fit
```

Call:

```
lm(formula = SkinDamage ~ SunScreen * SunExposure, data = dd)
```

Coefficients:

(Intercept)	SunScreen	SunExposure
4	-1	5
SunScreen:SunExposure		
-2		

```
wald(fit, 'SunScreen')    # test whether we can drop SunScreen altogether
```

	numDF	denDF	F-value	p-value
SunScreen	2	136	41.875	<.00001

	Estimate	Std.Error	DF	t-value	p-value	Lower	Upper
SunScreen	-1	0.338062	136	-2.958040	0.00365	-1.668538	0.95
SunScreen:SunExposure	-2	0.484031	136	-4.131969	0.00006	-2.957200	0.95
SunScreen	-0.331462						
SunScreen:SunExposure	-1.042800						

```
wald(fit, 'SunExposure')    # test whether we can drop SunExposure altogether
```

	numDF	denDF	F-value	p-value
SunExposure	2	136	146.875	<.00001

	Estimate	Std.Error	DF	t-value	p-value	Lower	Upper
SunExposure	5	0.338062	136	14.790199	<.00001	4.331462	0.95
SunScreen:SunExposure	-2	0.484031	136	-4.131969	6e-05	-2.957200	0.95

```
SunExposure          5.668538
SunScreen:SunExposure -1.042800
```

```
wald(fit, ':') # test whether we can drop all interaction terms (useful if many)
```

```
      numDF denDF  F-value p-value
:         1   136 17.07317   6e-05

              Estimate Std.Error DF  t-value  p-value Lower 0.95
SunScreen:SunExposure -2         0.484031  136 -4.131969 6e-05   -2.9572
              Upper 0.95
SunScreen:SunExposure -1.0428
```

3 The Paik-Agresti Diagram

Two diagrams that summarize all

- The Paik-Agresti Diagram
- Small project: Add the Liu-Meng marginal effect of Z line

UCBAdmissions

, , Dept = A

Admit	Gender	
	Male	Female
Admitted	512	89
Rejected	313	19

, , Dept = B

Admit	Gender	
	Male	Female
Admitted	353	17
Rejected	207	8

, , Dept = C

Gender

Admit	Male	Female
Admitted	120	202
Rejected	205	391

, , Dept = D

	Gender	
Admit	Male	Female
Admitted	138	131
Rejected	279	244

, , Dept = E

	Gender	
Admit	Male	Female
Admitted	53	94
Rejected	138	299

, , Dept = F

	Gender	
Admit	Male	Female
Admitted	22	24
Rejected	351	317

```
UCBAdmissions %>% as.data.frame
```

	Admit	Gender	Dept	Freq
1	Admitted	Male	A	512
2	Rejected	Male	A	313
3	Admitted	Female	A	89
4	Rejected	Female	A	19
5	Admitted	Male	B	353
6	Rejected	Male	B	207
7	Admitted	Female	B	17
8	Rejected	Female	B	8
9	Admitted	Male	C	120
10	Rejected	Male	C	205
11	Admitted	Female	C	202

12	Rejected	Female	C	391
13	Admitted	Male	D	138
14	Rejected	Male	D	279
15	Admitted	Female	D	131
16	Rejected	Female	D	244
17	Admitted	Male	E	53
18	Rejected	Male	E	138
19	Admitted	Female	E	94
20	Rejected	Female	E	299
21	Admitted	Male	F	22
22	Rejected	Male	F	351
23	Admitted	Female	F	24
24	Rejected	Female	F	317

```
UCBAdmissions %>% as.data.frame -> ducb
```

```
fit <- glm(Admit ~ Gender, ducb, family = binomial, weights = Freq)
summary(fit)
```

Call:

```
glm(formula = Admit ~ Gender, family = binomial, data = ducb,  
     weights = Freq)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.22013	0.03879	5.675	1.38e-08 ***
GenderFemale	0.61035	0.06389	9.553	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 6044.3 on 23 degrees of freedom
Residual deviance: 5950.9 on 22 degrees of freedom
AIC: 5954.9

Number of Fisher Scoring iterations: 4

```
fitbydept <- glm(Admit ~ Dept/Gender - 1, ducb, family = binomial, weights = Freq)
summary(fitbydept)
```

Call:

```
glm(formula = Admit ~ Dept/Gender - 1, family = binomial, data = ducb,
     weights = Freq)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)	
DeptA	-0.49212	0.07175	-6.859	6.94e-12	***
DeptB	-0.53375	0.08754	-6.097	1.08e-09	***
DeptC	0.53552	0.11494	4.659	3.18e-06	***
DeptD	0.70396	0.10407	6.764	1.34e-11	***
DeptE	0.95696	0.16160	5.922	3.18e-09	***
DeptF	2.76974	0.21978	12.602	< 2e-16	***
DeptA:GenderFemale	-1.05208	0.26271	-4.005	6.21e-05	***
DeptB:GenderFemale	-0.22002	0.43759	-0.503	0.615	

DeptC:GenderFemale	0.12492	0.14394	0.868	0.385
DeptD:GenderFemale	-0.08199	0.15021	-0.546	0.585
DeptE:GenderFemale	0.20019	0.20024	1.000	0.317
DeptF:GenderFemale	-0.18890	0.30516	-0.619	0.536

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

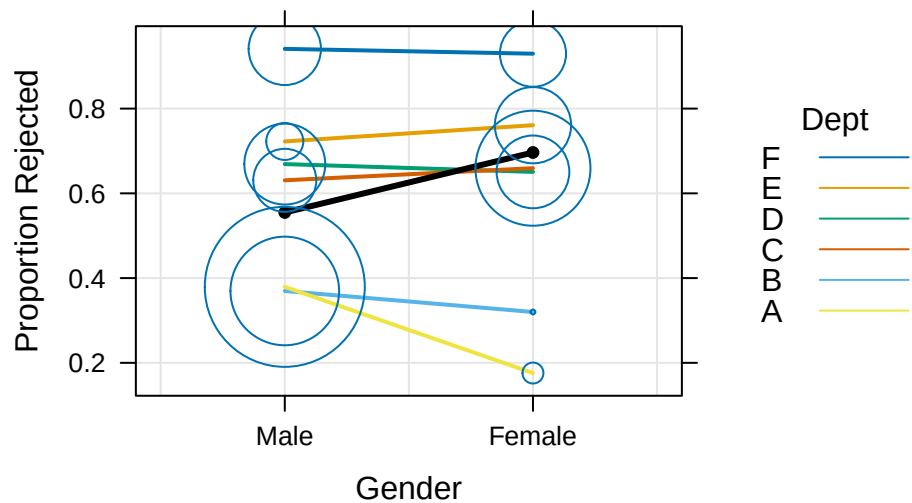
(Dispersion parameter for binomial family taken to be 1)

Null deviance: 6274.4 on 24 degrees of freedom
Residual deviance: 5167.3 on 12 degrees of freedom
AIC: 5191.3

Number of Fisher Scoring iterations: 6

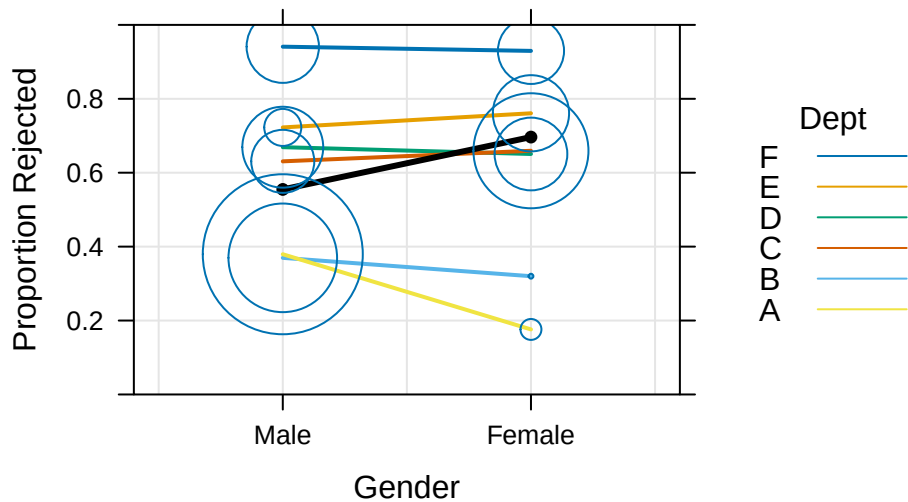
```
library(spida2)
```

```
paik(Admit ~ Gender + Dept | Freq, data = ducb,  
      ylab = 'Proportion Rejected')
```



Adjusting limits for the y-axis

```
paik(Admit ~ Gender + Dept | Freq, data = ducb,  
      ylab = 'Proportion Rejected', ylim = c(0,1))
```



4 Exercise 1: How a superficial model can hide inequity.

Consider the `death.penalty` data in the **spida2** package. It details the frequencies with which defendants received the death penalty (not necessarily carried out) in 674 homicide trials in the state of Florida from 1976 to 1987. The cases are classified according to the defendant's and the victim's race.

Using only the defendant's race as a predictor could suggest that the proportion of defendants receiving death sentences is relative similar between the two races.

Adjusting for the victims' race, however, suggests a very different and disturbing picture. This illustrates how a superficial analysis can fail to reveal important issues and lead to entirely different conclusions.

5 Exercise 2: Simpson's Paradox and Choosing a Restaurant: A hypothetical example (tricky!)

Mary (a woman) is choosing between restaurants A and B to take her friend, John (a man), out for dinner.

Restaurant A has an average rating of 4.1 and restaurant B of 4.3.

But looking at ratings by gender, among men, restaurant A has a rating of 4.0 and restaurant B of 3.8. Among women, restaurant A has a rating of 4.6 and restaurant B of 4.4. It seems that men and women separately prefer restaurant A but together they prefer restaurant B!

- 1) Draw a Paik-Agresti or other suitable diagram conditioning on gender to represent this data.
- 2) Draw a causal graph describing a plausible relationship among the three variables: restaurant, gender and ratings.
- 3) Without reference to possible causal mechanisms, explain numerically how the discrepancy between relative overall ratings and relative ratings by gender occurs.
- 4) Assuming that there are no other relevant factors related to restaurant ratings, what kind of variable would gender be in this context (you may now consider possible causal mechanisms)? Which restaurant should Mary choose? Why?
- 5) Now suppose that you have the same data but the variables are different. Restaurant A has an average rating of 4.1 and restaurant B of 4.3, as before. Each restaurant has two waiters who, very coincidentally, happen to have the same two names in each restaurant: Mr. Good and Mr. Bad. Among patrons served Mr. Bad in restaurant A, the ratings are 4.0 and those served by Mr. Bad (different waiter but the same name) in restaurant B, the ratings are 3.8.

Among patrons served Mr. Good in restaurant A, the ratings are 4.6 and those served by Mr. Good in restaurant B, the ratings are 4.4.

Without reference to possible causal mechanisms, explain numerically how such a discrepancy arises.

- 6) Assuming that there are no other relevant factors related to restaurant ratings and that the propensity for diners to report a rating is independent of waiters, what kind of variable would 'waiter' be in this context (you may now consider possible causal mechanisms)? Which restaurant should Mary choose? Why?

6 The Liu-Meng Diagram: The main effect of Z

You could always flip the roles of X and Z and draw a Paik-Agresti diagram for the main effect of Z.

But that wouldn't show the two in the same diagram.

The Liu-Meng diagram (personal communication over lunch with Xiao-Li Meng, Liu was his undergraduate student who worked on the idea with Xiao-Li. Unfortunately I don't know Liu's given name but someday I must find out.

1) Plot the points:

- (mean of X, mean of Y) | $Z = 0$
 - this is a weighted mean of the
 - * (mean of X, mean of Y) | $Z = 0, X = 0$, and
 - * (mean of X, mean of Y) | $Z = 0, X = 1$ Thus, it lies in the convex hull of these two points, which is the line joining the two points!
- (mean of X, mean of Y) | $Z = 1$
 - this is a weighted mean of the
 - * (mean of X, mean of Y) | $Z = 1, X = 0$, and
 - * (mean of X, mean of Y) | $Z = 1, X = 1$ Similarly, it also on the line joining these two

points.

- The vertical distance between these two points is the main effect of Z on Y.
- The horizontal distance between these two points is the main effect of Z on X.

Small project: Write a function that produces a Liu-Meng diagram.

R makes complicated things easy and sometimes easy things complicated:

```
ddZ <- spida2::up(dd, ~ SunExposure, agg = ~ SkinDamage + SunScreen)
ddZ
```

	SunExposure	Freq	SkinDamage	SunScreen
0	0	80	3.875	0.1250000
1	1	60	6.500	0.8333333

```
pull +
```

```
xyplot(SkinDamage ~ SunScreen, ddZ, type = 'b', lwd = 2, pch = 16, cex = 2)
```

