Graphing Longitudinal Data

Andrew Zieffler

Plotting Longitudinal Data

- Plots of change curves for individuals and aggregates
- Serve different purposes in exploratory and confirmatory analysis
- Use **ggplot2** package
- Use long format of data

Components of Plotting in **ggplot2**

- Based on Wilkinson's *Grammar of Graphics*
- Four components:
  - Aesthetic mappings
  - Geometric objects
  - Statistical transformations
  - Faceting

Aesthetic Mappings

- How variables are mapped to graph features
  - Define *x*-axis (time predictor) and *y*-axis (response)
  - Associate subjects with repeated measures
  - Aggregate individuals based on static predictors
- Specified with `aes()`

Geometric Objects

- Features drawn on plot (e.g., lines, points)
- Specified using prefix `geom_` and suffix that names feature to be plotted
  - Points specified with `geom_point()`
  - Lines specified with `geom_line()`

Statistical Transformations

- Used for plotting statistics (e.g., means)
  - Mean of the response at fixed levels of the predictor
- Specified using prefix `stat_` and suffix that names desired transformation
  - Means, medians, and other summary statistics specified with `stat_summary()`
  - Regression models specified with `stat_smooth()`

Faceting

- Creates separate plot for each subject or groups of subjects
  - Change curve for each subject specified with `facet_wrap()`
  - Facets based on values of static predictors specified with `facet_grid()`

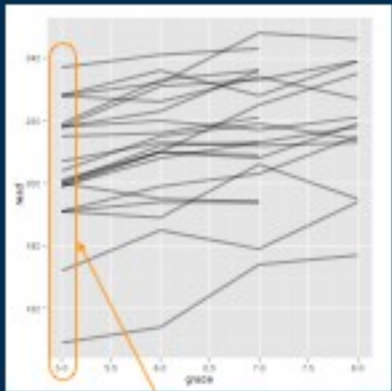
Plots are Built by Layering

- Plots are built in layers
- Each layer is a *sum* of components saved to an object
- First component of first layer is `ggplot()`
  - Contains reference to data frame and aesthetic mapping

Plotting Individual Change Curves

- Spaghetti plot
- All change curves superimposed on plot
- Long format data used
- Rows with missing values omitted

```
> library( ggplot2 )

> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line()
```

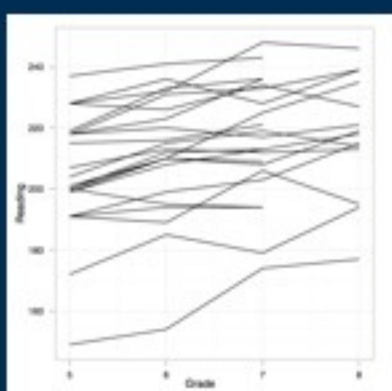


Individual variation in start points

- Most subjects increase reading score over time (some decrease)
- Does not appear linear

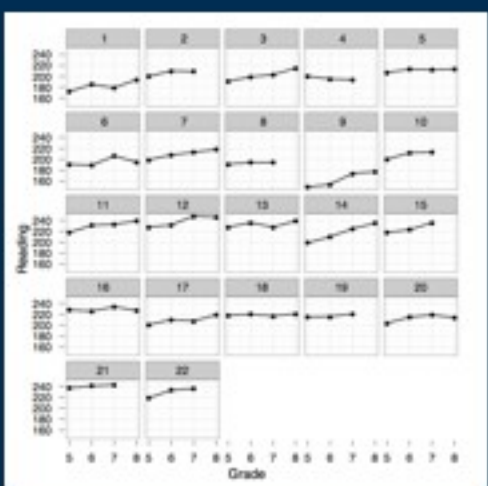
- No outlying subjects
- No outlying observations

```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line() +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" )
```



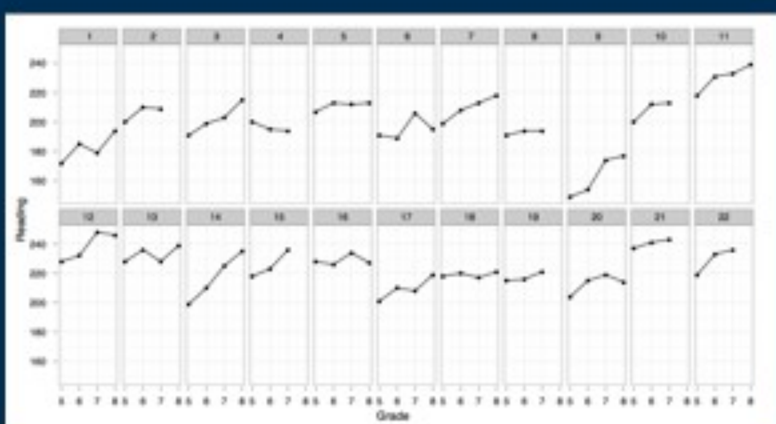
Facet Plots

```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line() +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" )
```



- Easier to see individual change curves
- Can see missing data (ID 2)

```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line() +
    geom_point() +
    facet_wrap( ~ subid, nrow = 2 ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" )
```



Plotting Subsets of Subjects

- Use the `subset()` function

logical expression

```
> sub1 <- subset( mpls.l, subid < 6 )
```

select subjects 1, 2, 3, 4, and 5

```
> ggplot( data = sub1, aes( x = grade, y = read, group = subid ) ) +
    geom_line() +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" )
```

- Within `subset()` can use more complex selection criterion
- `%in%` used to select a subset of subjects
- Useful for plotting random samples of subjects

```
> samp <- sample( 1:22, size = 4 )
```

sample 4 subjects between 1 and 22

```
> sub2 <- subset( mpls.l, subid %in% samp )
```

use those subjects

```
> ggplot( data = sub2, aes( x = grade, y = read, group = subid ) ) +
    geom_line() +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" )
```

Plotting Fitted Curves

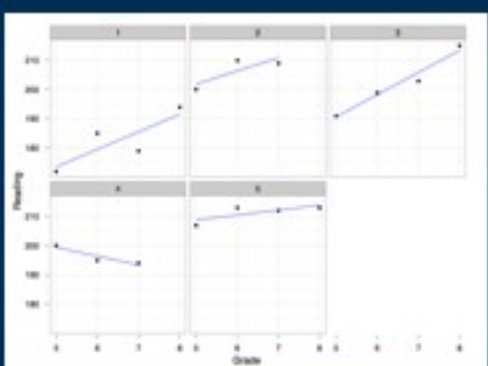
- Change curves have been fitted by connecting observed values
- Observed values contain measurement error
- Change curves are likely too specific
- Better to consider summary-based curves

OLS Fitted Curves

- OLS based summary curves
- Estimated by including `method="lm"` in the `stat_smooth()` function
- Remove `geom_line()` component

```
> ggplot( data = sub1, aes( x = grade, y = read, group = subid ) ) +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    stat_smooth( method = "lm", se = FALSE )
```

Turn off confidence envelopes



- Fitted OLS curves based on `read~grade`
- Most panels consistent with idea that observed reading score vary randomly around fitted line

OLS Fitted Curves

- Often reasonable to consider higher order polynomials for relationship between response and time predictor
- Most common higher order polynomials in social sciences are quadratic and cubic
- Estimated by including `formula=y~poly(x,p)` in the `stat_smooth()` function, where `p` is `2` for quadratic and `3` for cubic

Linear

$$\hat{y} = \beta_0 + \beta_1(\text{grade})$$

Quadratic

$$\hat{y} = \beta_0 + \beta_1(\text{grade}) + \beta_3(\text{grade}^2)$$

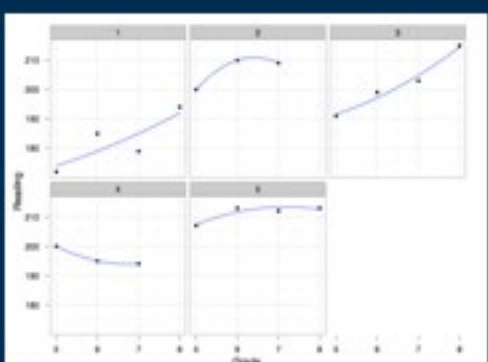
Cubic

$$\hat{y} = \beta_0 + \beta_1(\text{grade}) + \beta_3(\text{grade}^2) + \beta_4(\text{grade}^3)$$

These are *raw* or *correlated* polynomials. By default `poly()` creates *orthogonal* or *uncorrelated* polynomials. Including the argument `raw=TRUE` will create raw polynomials.

Quadratic polynomial

```
> ggplot( data = sub1, aes( x = grade, y = read, group = subid ) ) +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    stat_smooth( method = "lm", se = FALSE, formula = y ~ poly( x, 2) )
```



- Polynomial for subject 1 and 3 do not deviate much from linear
- Curves for subject 2 and 4 indicate perfect fit!?

Saturation

- The number of parameters in the LM must be fewer than the number of time points
- For example, consider linear curve with 2 parameters. Line fits perfectly with 2 points.
- Saturated models do not summarize well and should be avoided

Cubic polynomial

```
> ggplot( data = sub1, aes( x = grade, y = read, group = subid ) ) +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    stat_smooth( method = "lm", se = FALSE, formula = y ~ poly( x, 3) )

Error in poly(x, 3) : 'degree' must be less than number of unique points
```

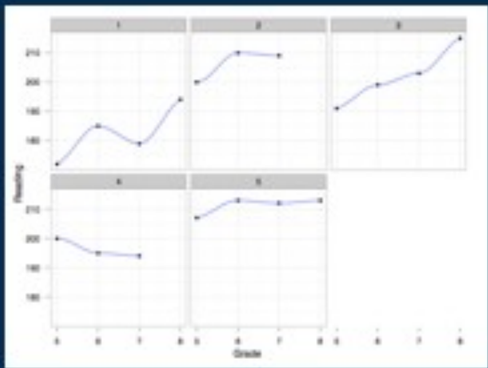
Since many subjects only have 3 observations, error is produced. To avoid this, select subset of subjects with 4 observations (no missing values).

Fitted Curves using Local Smoothing

- Alternative to OLS polynomials
- Estimated by not including `method=` in the `stat_smooth()` function
- More useful at group levels since it is based on density
- If too few observations, error will be produced

Local smoothing

```
> ggplot( data = sub1, aes( x = grade, y = read, group = subid ) ) +
    geom_point() +
    facet_wrap( ~ subid ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    stat_smooth( se = FALSE )
```



- Observed values connected, but more "wavy"

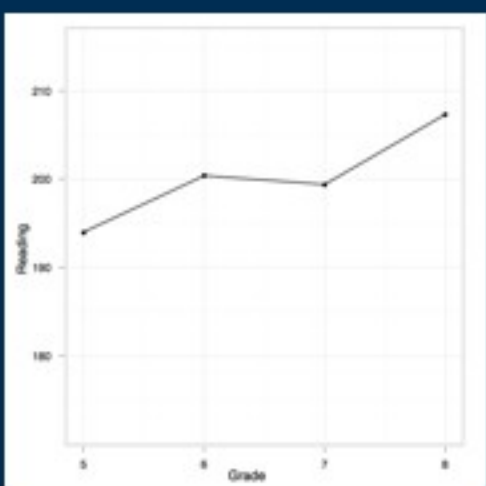
Plotting Group Level Curves

- Mean curve
- Fixed effects part of LMER models mean change over time
- Omit `group=` from `aes()` component. Also omit `facet_wrap()` component
- Include `fun.y=mean` and `geom="line"` in `stat_summary()` component

Mean curve

```
> ggplot( data = mpls.l, aes( x = grade, y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    stat_summary( fun.y = mean, geom = "line" ) +
    stat_summary( fun.y = mean, geom = "point" )
```

Add points at means



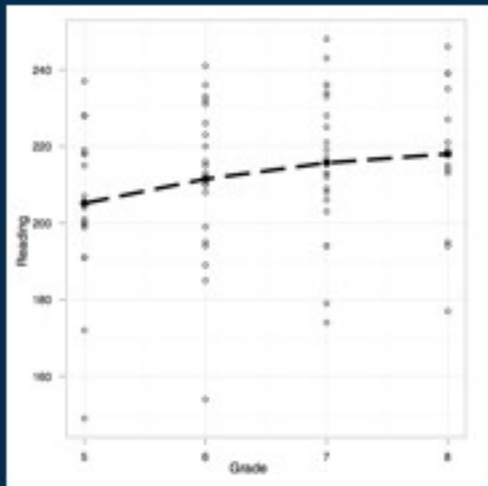
- Mean change shows growth
- Also shows deceleration (non-linear change)

Mean Change Curve

- Include geom_point() component to show individual deviation from mean curve
- Line type can be changed by adding lty=
- Line width can be changed by adding lwd=
- Point type can be changed by adding pch=
- Point size can be changed by adding cex=

Mean curve

```
> ggplot( data = mpls.l, aes( x = grade, y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    stat_summary( fun.y = mean, geom = "line", lwd = 1.5, lty = 5 ) +
    stat_summary( fun.y = mean, geom = "point", pch = 19, cex = 3 ) +
    geom_point( pch = 1 )
```



Unbalanced Data

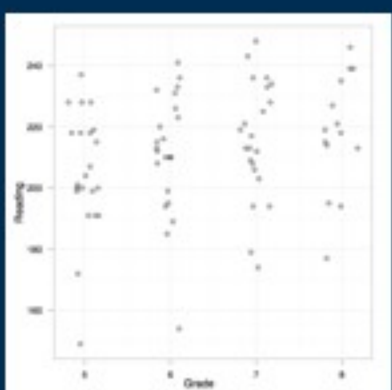
- Data are not balanced on time
- Can't tell this on plot because points are overplotted
- Add a small amount of variation to (x, y) coordinates when plotting to "spread out" points so overplotting does not occur by using `jitter()`

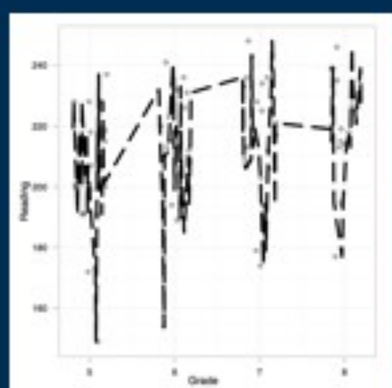
Jittering points

```
> ggplot( data = mpls.l, aes( x = jitter( grade ), y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    geom_point( pch = 1 )
```

Jittering points with mean curve superimposed

```
> ggplot( data = mpls.l, aes( x = jitter( grade ), y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    geom_point( pch = 1 ) +
    stat_summary( fun.y = mean, geom = "line", lwd = 1.5, lty = 5 )
```





Fitted Group Level Curves

- Same syntax as for individual curves, but `group=` is omitted from `aes()` component
- Use `opts()` component to set aspect ratio of plot to produce square graph using `"aspect.ratio"=1` (less visual distortion)

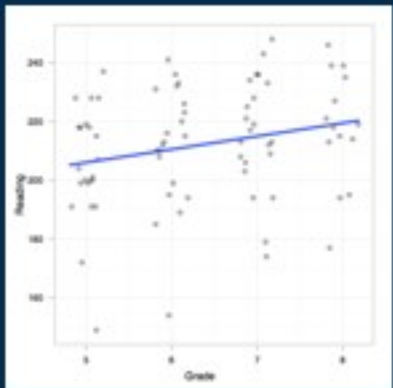
Regression group level curve (linear)

```
> ggplot( data = mpls.l, aes( x = jitter( grade ), y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    geom_point( pch = 1 ) +
    stat_smooth( method = "lm", se = FALSE, lwd = 1.5 ) +
    opts ( "aspect.ratio" = 1 )
```

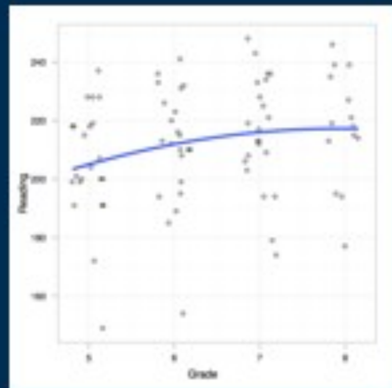
Regression group level curve (quadratic)

```
> ggplot( data = mpls.l, aes( x = jitter( grade ), y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    geom_point( pch = 1 ) +
    stat_smooth( method = "lm", se = FALSE, lwd = 1.5,
      formula = y ~ poly( x, 2 ) ) +
    opts ( "aspect.ratio" = 1 )
```

Linear

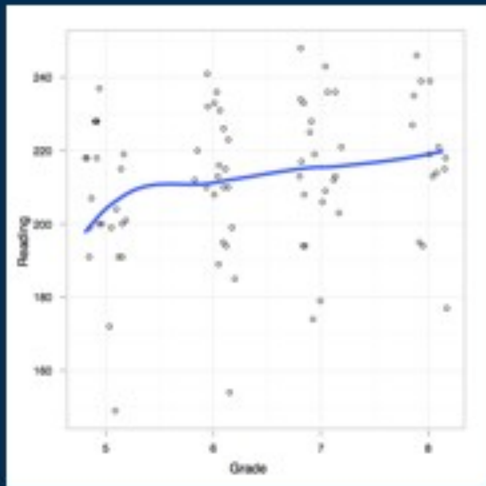


Quadratic



Local smoothed group level curve

```
> ggplot( data = mpls.l, aes( x = jitter( grade ), y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    geom_point( pch = 1 ) +
    stat_smooth( se = FALSE, lwd = 1.5 ) +
    opts ( "aspect.ratio" = 1 )
```



Local smoothed group level curve

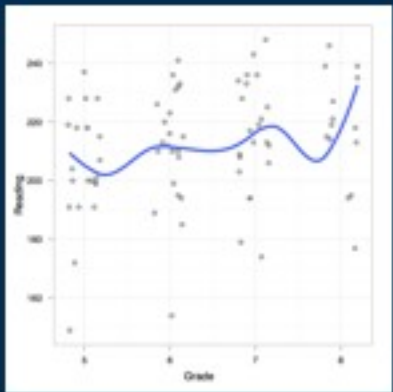
```
> ggplot( data = mpls.l, aes( x = jitter( grade ), y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    geom_point( pch = 1 ) +
    stat_smooth( se = FALSE, lwd = 1.5, span = 0.9) +
    opts ( "aspect.ratio" = 1 )
```

Smoothing parameter

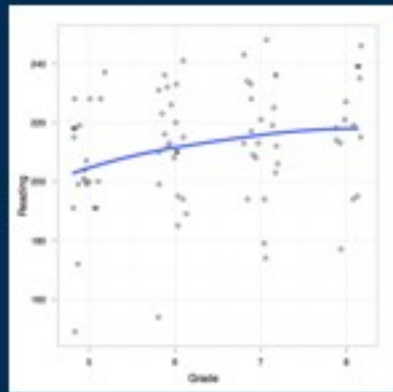
higher values = more smooth

default span=0.75

span=0.4



span=0.9



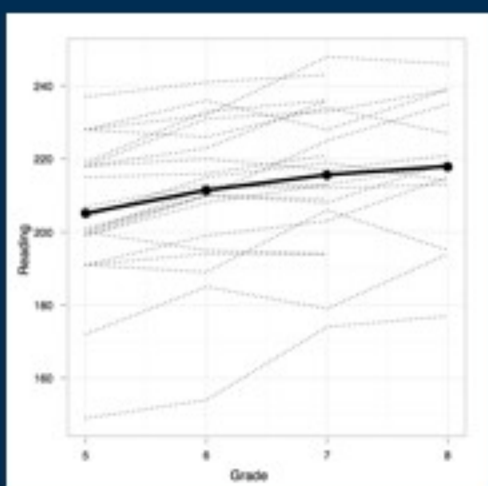
Group and Individual Level Curves

- Shows mean trend and individual variation
- Multiple use of `group=` in `aes()` component
- Use `group=subid` for individual curves
- Use `group=1` for mean curve

Individual change curves

```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line( lty = 3 ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "point",
      cex = 4 ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "line",
      lwd = 1.5 )
```

Group mean change curves



Conditioning on Static Predictors

- So far have considered *unconditional* change
- When static predictors are important covariates, should create plots for each level (i.e., examine change across levels of the static predictor)
- Plots can be superimposed on same graph or faceted

Categorical Static Predictors

- Superimposed group curves drawn by providing static predictor to `group=` in `stat_summary()` component
- Use different line types and plotting characters (points) for separate groups

Examine group sizes

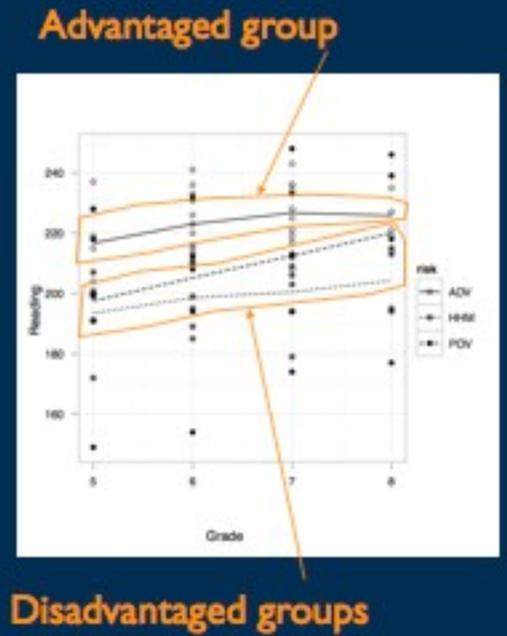
```
> table( mpls.l$risk )

ADV HHM POV
 48  24  24
```

Different point types for each group

```
> ggplot( data = mpls.l, aes( x = grade, y = read, shape = risk ) ) +
    geom_point() +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( line = risk ), fun.y = mean, geom = "line" ) +
    scale_shape_manual( values = c( 1, 8, 19 ) )
```

Different line types for each group

Choose point types



- Advantaged group shows higher mean reading scores at each time point
- Growth curves are different for the groups

Interactions

- Use `:` in `line=` or `shape=` in `stat_summary()` component

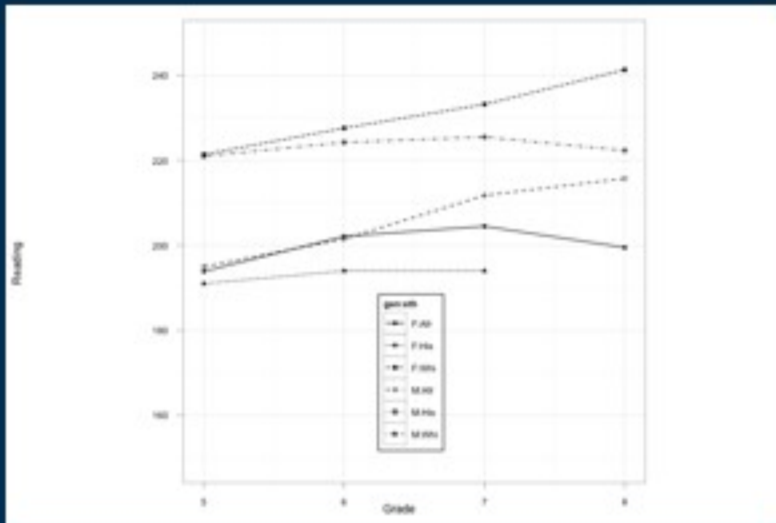
Examine group sizes

```
> table( mpls.l$gen, mpls.l$eth )

    Afr His Whi
  F  32   4  20
  M  16   0  16
```

Will not be plotted

Position the legend and add rectangle around it

```
> ggplot( data = mpls.l, aes( x = grade, y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1, legend.position = c( 0.52, 0.27 ),
      legend.background = theme_rect() ) +
    stat_summary( aes( line = gen : eth ), fun.y = mean, geom = "line" ) +
    stat_summary( aes( shape = gen : eth ), fun.y = mean,
      geom = "point", cex = 2 )
```

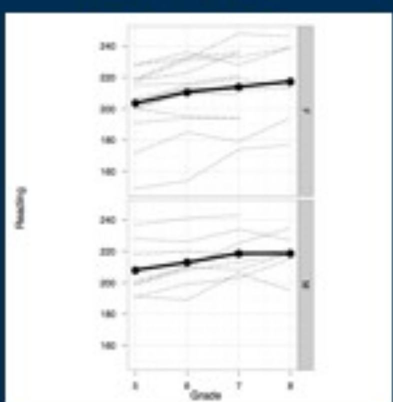


Faceting

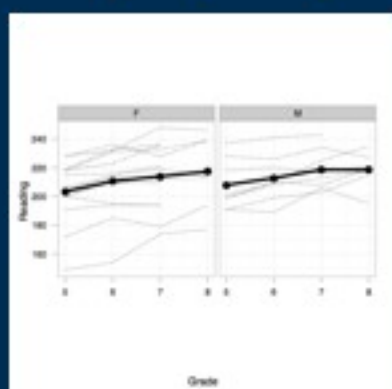
- Tilde (~) notation used within `facet_grid()` component
  - Row faceting before tilde
  - Column faceting after tilde
  - Period indicates no faceting
- `facet.grid( gen ~ . )` will facet rows by gender

```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line( lty = 3 ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "point" ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "line" ) +
    facet_grid( gen ~ . )
```

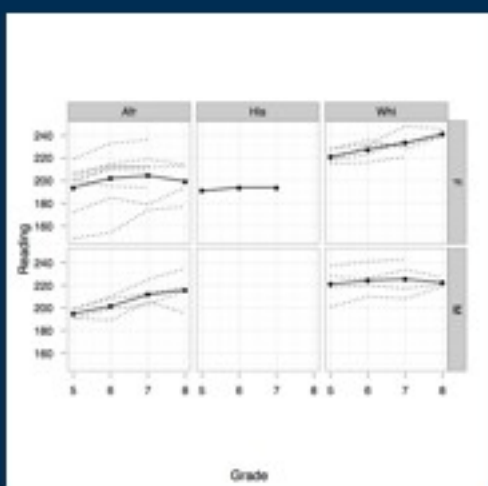
facet_grid(gen ~ .)



facet_grid(. ~ gen)



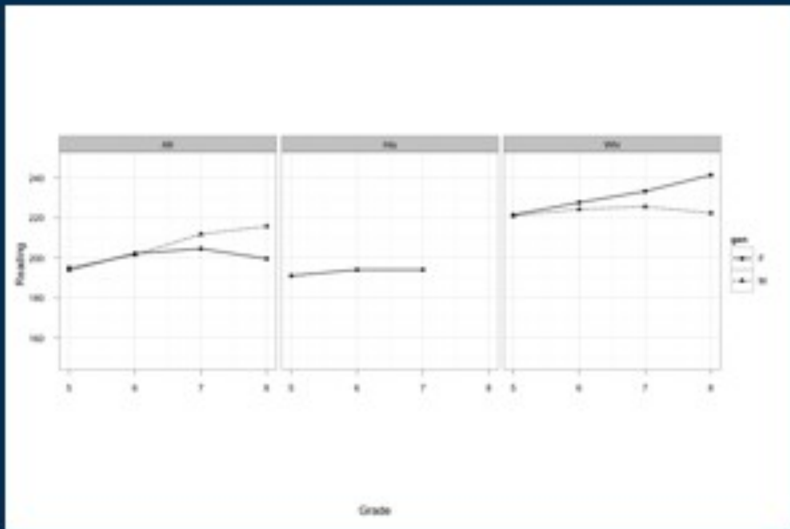
```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line( lty = 3 ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "point" ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "line" ) +
    facet_grid( gen ~ eth )
```



Superimposing and Faceting

- Multiple static predictors

```
> ggplot( data = mpls.l, aes( x = grade, y = read ) ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( line = gen ), fun.y = mean, geom = "line" ) +
    stat_summary( aes( shape = gen ), fun.y = mean, geom = "point" ) +
    facet_grid( . ~ eth )
```



Quantitative Predictors

- Typically no natural groups
- Create groups based on quantitative values
- Debate about validity (Gelman & Park, 2008; McClelland & Irwin, 2003)
- Grouping accomplished with `cut_interval()` and `cut_number()` functions

Quantitative Predictors

- `cut_interval()` creates groups of equal interval lengths based on argument `n=`
- `cut_number()` creates groups of equal size (unequal interval lengths) based on argument `n=`
- Better illustrated with more subjects

Simulate 100 values from ~$N(100, 15)$

```
> set.seed( 123 )
> x <- rnorm( n = 100, mean = 100, sd = 15 )

> table( cut_interval( x, n = 4 ) )

[65.4,82.2] (82.2,99.1]  (99.1,116]   (116,133]
          7          39          39          15

> table( cut_number( x, n = 4 ) )

[65.4,92.6]  (92.6,101]   (101,110]   (110,133]
         25          25          25          25
```

Median Split

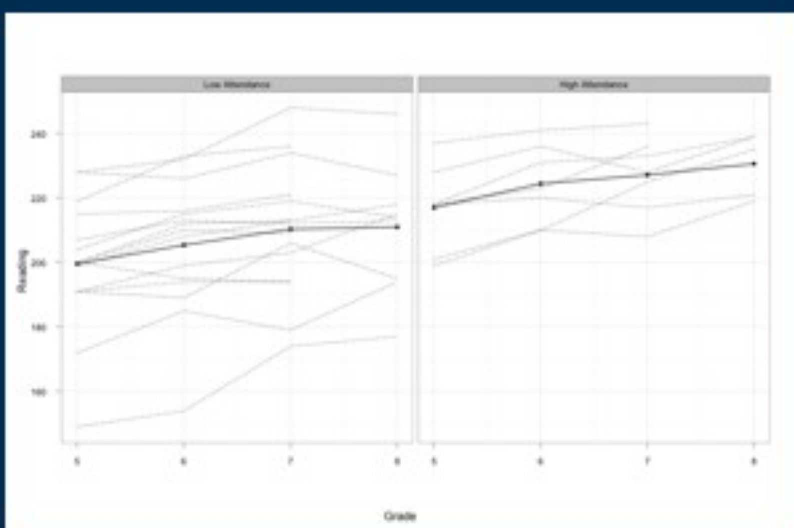
- Consider `att` variable, median = 97
- Use `cut_number()` to create two groups
- Assign these into new variable in the `mpls.l` data frame
- Both `cut_interval()` and `cut_number()` create factors with level information

Double-check that the function works

```
> table( cut_number( mpls.l$att, n = 2 ) )

[0.85,0.97]    (0.97,1]
         60          28
```

```
> mpls.l$att2 <- cut_number( mpls.l$att, n = 2 )

> levels( mpls.l$att2 ) <- c( "Low Attendance", "High Attendance" )
```

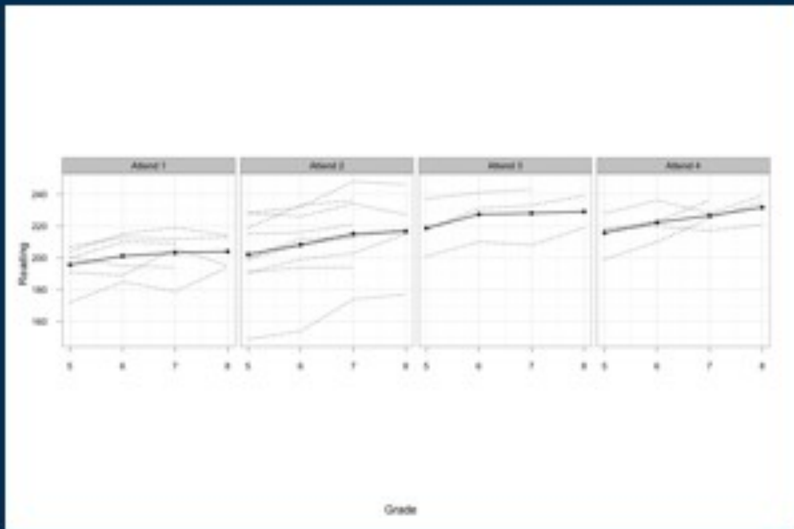
```
> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line( lty = 3 ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "line" ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "point" ) +
    facet_grid( . ~ att2 )
```



Two Attendance Groups

- Low attendance group starts at lower reading level (on average)
- Both groups increase over time (non-linear)
- More variation (vertical spread) in change curves at lower attendance level
- Variation in original variable truncated

```
> mpls.l$att2 <- cut_number( mpls.l$att, n = 4 )

> levels( mpls.l$att2 ) <- c( "Attend 1", "Attend 2", "Attend 3",
    "Attend 4" )

> ggplot( data = mpls.l, aes( x = grade, y = read, group = subid ) ) +
    geom_line( lty = 3 ) +
    theme_bw() +
    scale_x_continuous( name = "Grade", breaks = 5:8 ) +
    scale_y_continuous( name = "Reading" ) +
    opts ( "aspect.ratio" = 1 ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "line" ) +
    stat_summary( aes( group = 1 ), fun.y = mean, geom = "point" ) +
    facet_grid( . ~ att2 )
```



Two vs Four Groups

- Starting level varies based on attendance
- Lowest two levels have roughly the same starting values
- Recommendation: To thoroughly investigate effects of quantitative predictors, use ≥ 4 groups

References

Gelman, A., & Park, D. K. (2008). Splitting a predictor at the upper quarter or third and the lower quarter or third. *American Statistician*, *62*, 1–8. http://www.stat.columbia.edu/~gelman/research/published/thirds5.pdf

Irwin, J. R., & McClelland, G. H. (2003). Negative consequences of dichotomizing continuous predictor variables. *Journal of Marketing Research*, *40*, 366–371.

Further Reading

Wickham, H. (2009). *ggplot2: Elegant graphics for data analysis*. New York: Springer.