The Delta Method and Applications to Mark Recapture Models

Iowa State University, Ecology 607

November 16, 2020

Katherine Goode (kgoode@iastate.edu)

Find slides at: goodekat.github.io/presentations.html

Background: AES stats consulting

- Offer help with data analyses for ISU researchers (includes graduate students, post docs, faculty, and staff)
- Consulting website: https://www.stat.iastate.edu/statistical-consulting

Slide structure

1. What is the delta method?
2. Mathematical details
3. Delta method in R by hand
4. Delta method in R via `msm`
5. A more complicated example

What is the delta method?

Computing a confidence interval

Suppose you take measurements of the length of 30 frogs and want to compute:

- mean length
- standard deviation of lengths
- confidence interval for the mean

What do you need to compute a confidence interval for a mean?

Standard error

Standard error: estimate of variability associated with a statistic

Example:

Confidence interval for a sample mean:

$$\bar{y} \pm z \cdot \mathrm{SE}\left[\bar{y}\right]$$

where $\bar{y}$ is the sample mean, z is a normal quantile, and

$$\mathrm{SE}\left[\bar{y}\right] = \mathrm{sd}\left[\bar{y}\right] = \sqrt{Var\left[\bar{y}\right]} = \frac{s}{\sqrt{n}}$$

where s is the sample standard deviation and n is the sample size.

Computing standard errors

Some standard errors are easy to derive/compute (like a sample mean)

$$
\begin{aligned}
SE\left[\bar{y}\right] &= \sqrt{Var\left[\frac{1}{n}\sum_{i=1}^{n} y_i\right]} \\
&= \sqrt{\left(\frac{1}{n}\right)^2 \left(\sum_{i=1}^{n} Var\left[y_i\right]\right)} \\
&= \sqrt{\frac{1}{n^2}\left(nVar\left[y_i\right]\right)} \\
&= \sqrt{\frac{1}{n}Var\left[y_i\right]} \\
&= \frac{\sqrt{Var[y_i]}}{\sqrt{n}} \\
&= \frac{s}{\sqrt{n}}
\end{aligned}
$$

Others do not...

Estimating tricky standard errors

When computing "tricky" standard errors, it may be helpful to use the...

> Delta method: approach to approximate standard errors of transformed parameters If

- want a SE for a parameter estimate
- doesn't have a commonly used formula, is accessible via software, or derived easily

and

- parameter estimate is a function of other parameters with known SEs
- some other assumptions are met (to be discussed)

then

- delta method can be used to compute the standard error

Mule deer example

Suppose you have data on mule deer survival and calculate several quantities:

(1) Logistic regression equation in MARK for quarterly survival:

$$\log\left(\frac{S}{1-S}\right) = \beta_0 + \beta_1 \cdot \mathbf{age} + \beta_2 \cdot \mathbf{age}^2$$

where S represents the quarterly survival for a given $\mathbf{age}$

(2) Quarterly survival estimates for specific ages:

$$S = \frac{\exp\left(\beta_0 + \beta_1 \cdot \mathbf{age} + \beta_2 \cdot \mathbf{age}^2\right)}{1 + \exp(\beta_0 + \beta_1 \cdot \mathbf{age} + \beta_2 \cdot \mathbf{age}^2)}$$

(3) Annual survival for a specific age:

$$(S)^4$$

Try it out

Suppose that you are interested in obtaining confidence intervals for the $\hat{\boldsymbol{\beta}}$'s, $\hat{S}$ and $(\hat{S})^4$. Which would you need to use the delta method to derive?

- $\hat{\boldsymbol{\beta}}$'s
- $\hat{S}$
- $(\hat{S})^4$

Try it out: Solution

Suppose that you are interested in obtaining confidence intervals for the $\hat{\boldsymbol{\beta}}$'s, $\hat{S}$ and $(\hat{S})^4$. Which would you need to use the delta method to derive?

- $\hat{\boldsymbol{\beta}}$'s - No
- $\hat{S}$ - Can get from MARK but is (probably) derived using the delta method
- $(\hat{S})^4$ - Will need to derive the standard error using delta method

Summary

- Standard errors estimate variability associated with a statistic
- Standard errors are often used to compute confidence intervals
- Delta method can be used to approximate standard errors of transformed parameters that are not easy to compute

Mathematical details

Notation

Value	Definition
$\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$	parameter vector of length $\boldsymbol{p}$
$\hat{\boldsymbol{\theta}}$	sequence of estimators of $\boldsymbol{\theta}$
$g(\boldsymbol{\theta})$	function of $\boldsymbol{\theta}$
$g\left(\hat{\boldsymbol{\theta}}\right)$	estimate of $\boldsymbol{g(\theta)}$
$\mathbf{d}$	vector of partial derivatives of length $\boldsymbol{p}$ with a $\boldsymbol{j}$th element of $$\frac{\partial g(\boldsymbol{\theta})}{\partial \theta_j}$$

Try it out

Want to compute standard error for S^4 where

$$S = \frac{\exp\left(\beta_0 + \beta_1 \cdot \mathrm{age} + \beta_2 \cdot \mathrm{age}^2\right)}{1 + \exp(\beta_0 + \beta_1 \cdot \mathrm{age} + \beta_2 \cdot \mathrm{age}^2)}$$

What is the notation in this context?

- $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$
- $\hat{\boldsymbol{\theta}}_n$
- $g(\boldsymbol{\theta})$
- $g\left(\hat{\boldsymbol{\theta}}\right)$
- $\mathbf{d}$

Try it out: Solution

Want to compute standard error for S^4 where

$$S = \frac{\exp\left(\beta_0 + \beta_1 \cdot \text{age} + \beta_2 \cdot \text{age}^2\right)}{1 + \exp(\beta_0 + \beta_1 \cdot \text{age} + \beta_2 \cdot \text{age}^2)}$$

What is the notation in this context?

- $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p) = \boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)$
- $\hat{\boldsymbol{\theta}}_n = \hat{\boldsymbol{\beta}}_n = \left(\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2\right)$
- $g(\boldsymbol{\theta}) = S^4 = g\left(\boldsymbol{\beta}\right) = \left(\frac{\exp\left(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2\right)}{1+\exp\left(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2\right)}\right)^4$
- $g\left(\hat{\boldsymbol{\theta}}\right) = \hat{S}^4 = g\left(\hat{\boldsymbol{\beta}}_n\right) = \left(\frac{\exp\left(\hat{\beta}_0+\hat{\beta}_1\cdot\text{age}+\hat{\beta}_2\cdot\text{age}^2\right)}{1+\exp\left(\hat{\beta}_0+\hat{\beta}_1\cdot\text{age}+\hat{\beta}_2\cdot\text{age}^2\right)}\right)^4$
- $\mathbf{d} = \left(\frac{\partial g(\boldsymbol{\beta})}{\partial \beta_0}, \frac{\partial g(\boldsymbol{\beta})}{\partial \beta_1}, \frac{\partial g(\boldsymbol{\beta})}{\partial \beta_2}\right)$ (estimate partial derivatives in practice using $\hat{\boldsymbol{\beta}}$)

The Delta Method

Situation/conditions

- $\boldsymbol{\theta}$ has mean $\boldsymbol{\theta}$ and variance covariance matrix $Cov\left(\hat{\boldsymbol{\theta}}\right)$
- $g\left(\boldsymbol{\theta}\right)$ is a function of parameters $\boldsymbol{\theta}$ and is real-valued and continuously differentiable in a neighborhood of $\boldsymbol{\theta}$
- $\hat{\boldsymbol{\theta}}$ are asymptotically normal estimators of $\boldsymbol{\theta}$

Results

- $g\left(\hat{\boldsymbol{\theta}}\right)$ follows a normal distribution:

$$g\left(\hat{\boldsymbol{\theta}}\right) \sim N\left(g(\boldsymbol{\theta}),\ \mathbf{d}Cov\left(\hat{\boldsymbol{\theta}}\right)\mathbf{d}'\right)$$

- .Standard error of $g\left(\hat{\boldsymbol{\theta}}\right)$ can be computed as

$$SE\left(g\left(\hat{\boldsymbol{\theta}}\right)\right) = \sqrt{Cov\left(g\left(\hat{\boldsymbol{\theta}}\right)\right)} = \sqrt{\mathbf{d}Cov\left(\hat{\boldsymbol{\theta}}\right)\mathbf{d}'}.$$

(Note that $Cov\left(\hat{\boldsymbol{\theta}}\right)$ is estimated by the negative inverse Hessian matrix)

Computing mule deer standard error

Now, how to compute the standard error for $\hat{S}^4$?

$$SE\left(\hat{S}^4\right) = SE\left(g\left(\hat{\boldsymbol{\beta}}\right)\right) = \sqrt{\mathbf{d}Cov\left(\hat{\boldsymbol{\beta}}\right)\mathbf{d}'}$$

Would need to:

- Derive partial derivatives in $\mathbf{d}$ (see next slide)
- Compute partial derivatives using estimated parameters from the logistic regression model
- Extract variance-covariance matrix from logistic regression for $Cov\left(\hat{\boldsymbol{\beta}}\right)$
- Use formula above to put it all together for computing $SE\left(g\left(\hat{\boldsymbol{\beta}}\right)\right)$

Mule deer derivatives

The partial derivatives of $\boldsymbol{g}(\boldsymbol{\beta})$ in terms of $\boldsymbol{\beta_0}$, $\boldsymbol{\beta_1}$, and $\boldsymbol{\beta_2}$ (note that the partial derivatives are functions of age):

$$
\begin{aligned}
\mathbf{d}' &= \left[\begin{array}{ccc} \frac{\partial}{\partial\beta_0}(S^4) & \frac{\partial}{\partial\beta_1}(S^4) & \frac{\partial}{\partial\beta_2}(S^4) \end{array}\right]' \\
&= \left[\begin{array}{l}
\frac{\partial}{\partial\beta_0}\left(\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)^4 \\
\frac{\partial}{\partial\beta_1}\left(\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)^4 \\
\frac{\partial}{\partial\beta_2}\left(\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)^4
\end{array}\right] \\
&= \left[\begin{array}{l}
4\left(\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)^4\left(1-\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right) \\
4\left(\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)^4\left(1-\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)(\text{age}) \\
4\left(\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)^4\left(1-\frac{\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}{1+\exp(\beta_0+\beta_1\cdot\text{age}+\beta_2\cdot\text{age}^2)}\right)(\text{age}^2)
\end{array}\right]. \\
&= \left[\begin{array}{l}
4S^4(1-S) \\
4S^4(1-S)(\text{age}) \\
4S^4(1-S)(\text{age})^2
\end{array}\right]
\end{aligned}
$$

Summary

- For parameters $\hat{\boldsymbol{\theta}}$ with a known variance covariance matrix $Cov\left(\hat{\boldsymbol{\theta}}\right)$:
    - Can use the delta method to obtain a standard error for a transformation of the parameters: $g\left(\hat{\boldsymbol{\theta}}\right)$
- Delta method standard error formula:

$$SE\left(g\left(\hat{\boldsymbol{\theta}}\right)\right) = \sqrt{\mathbf{d}Cov\left(\hat{\boldsymbol{\theta}}\right)\mathbf{d}'}$$

- Involves computing partial derivatives

Delta method in R by hand

Mule deer example in R

MARK model results

```r
# R packages
library(dplyr); library(readr)
```

```r
# Load data frame of quarterly survival, quarterly survival,
# standard error of quarterly survival, and annual survival
# computed based on the logistic regression model fit in MARK
survival_data <- read_csv("data/deer_model_results.csv")
# Print the head of the data
head(survival_data)
```

```
## # A tibble: 6 x 5
##     age logit s_quarterly     se s_annual
##   <dbl> <dbl>       <dbl>  <dbl>    <dbl>
## 1   1     2.70       0.937 0.0219    0.771
## 2   1.1   2.73       0.939 0.0206    0.776
## 3   1.2   2.75       0.940 0.0193    0.780
## 4   1.3   2.77       0.941 0.0182    0.784
## 5   1.4   2.79       0.942 0.0172    0.788
## 6   1.5   2.81       0.943 0.0163    0.792
```

Mule deer logisitc regression estimates

Model coefficients

```
# Load logistic regression coefficient estimates
deer_betas <- read_csv("data/deer_betas.csv", col_names = FALSE) %>%
  as.matrix()
deer_betas
```

```
##              X1
## [1,]  2.4060341
## [2,]  0.3446497
## [3,] -0.0494241
```

Variance-covariance matrix

```
# Load estimated variance covariance matrix of the logistic regression 
deer_vcov <- read_csv("data/deer_vcov.csv", col_names = FALSE) %>%
  as.matrix()
deer_vcov
```

```
##               X1           X2            X3
## [1,]  0.33546317 -0.145305616  0.0129216170
## [2,] -0.14530562  0.083162139 -0.0082607889
## [3,]  0.01292162 -0.008260789  0.0008725783
```

Hand written R function

`compute_annual_se`: Function for computing annual survival standard error via the delta method

Inputs:

- `age`: age at which to compute annual survival estimate and standard error
- `betas`: estimated logistic regression coefficients (vector of length 3)
- `vcov`: estimated variance covariance matrix of logistic regression coefficients (3x3 matrix)

Output: data frame with the variables of...

- `age`: age specified for computations
- `annual_survival`: estimated annual survival for specified `age`
- `se`: standard error for annual survival (estimated using delta method)
- `lower`: lower bound of 95% confidence interval for annual survival
- `upper`: upper bound of 95% confidence interval for annual survival

```r
# Function for computing annual survival standard error
compute_annual_se <- function(age, betas, vcov){

  # Separate the betas
  b0 <- betas[1]; b1 <- betas[2]; b2 <- betas[3]

  # Compute logit of quarterly survival for given age
  logit_s <- b0 + (b1 * age) + (b2 * (age^2))

  # Compute quarterly and annual survival
  s <- exp(logit_s) / (1 + exp(logit_s))
  annual <- s^4

  # Create empty 1x3 matrix to store the elements of d
  d <- matrix(NA, nrow = 1, ncol = 3)

  # Compute elements of d (partial derivatives of g(beta))
  d[1] <- 4 * (s^4) * (1 - s)
  d[2] <- 4 * age * (s^4) * (1 - s)
  d[3] <- 4 * (age^2) * (s^4) * (1 - s)

  # Compute standard error of annual survival (using delta method)
  se <- sqrt(d %*% vcov %*% t(d))

  # Compute lower and upper bounds of 95% CI for annual survival
  lower <- annual - (1.96 * se); upper <- annual + (1.96 * se)

  # Return age, annual survival estimate, standard error, 95% CI
  return(data.frame(age, annual_survival = annual, se, lower, upper))
}
```

Applying the function to one age

```
# Apply compute_annual_se when age is 1
age1_se <-
  compute_annual_se(
    age = survival_data$age[1],
    betas = deer_betas,
    vcov = deer_vcov
  )
```

```
# Print the results
age1_se
```

```
##   age annual_survival         se     lower     upper
## 1   1       0.7711618 0.07213014 0.6297867 0.9125368
```

Applying the function to multiple ages

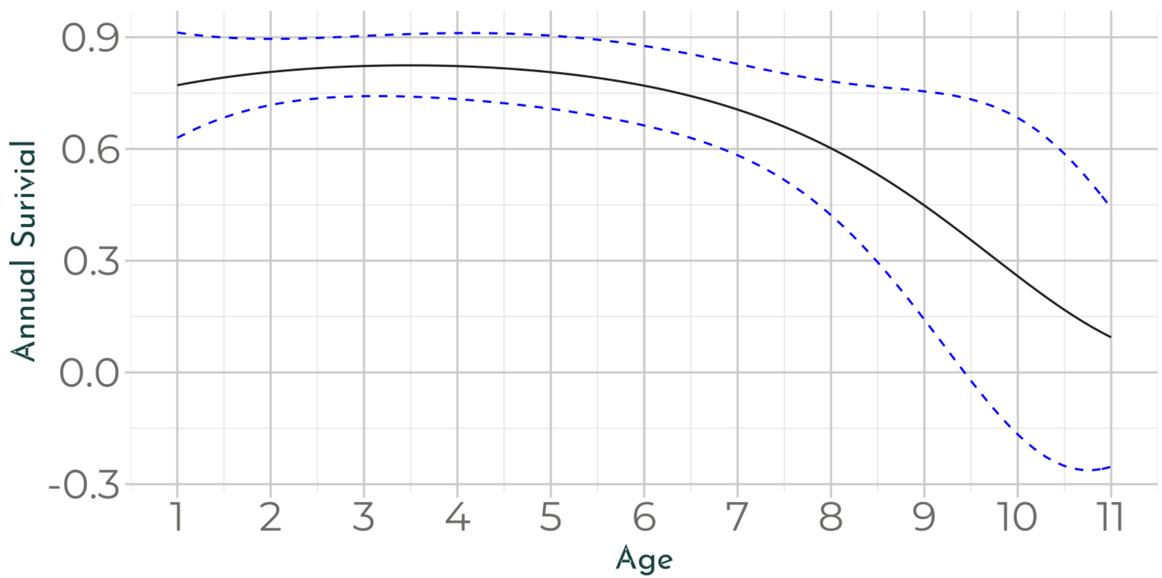
```r
# Apply compute_annual_se to all of the ages in survival data
# using map_df from purrr to apply .f to all elements of .x
survival_data_annual <-
  purrr::map_df(
    .x = survival_data$age,
    .f = compute_annual_se,
    betas = deer_betas,
    vcov = deer_vcov
  )
```

```r
# Print part of the results
head(survival_data_annual)
```

```
##   age annual_survival         se     lower     upper
## 1 1.0       0.7711618 0.07213014 0.6297867 0.9125368
## 2 1.1       0.7757964 0.06799871 0.6425190 0.9090739
## 3 1.2       0.7801689 0.06422088 0.6542960 0.9060418
## 4 1.3       0.7842886 0.06078494 0.6651502 0.9034271
## 5 1.4       0.7881646 0.05767921 0.6751134 0.9012159
## 6 1.5       0.7918054 0.05489182 0.6842175 0.8993934
```

Annual surival by age with confidence intervals

```
# Plot the estimated annual survival and confidence intervals
library(ggplot2)
ggplot(survival_data_annual, aes(x = age, y = annual_survival)) +
  geom_line() +
  geom_line(aes(x = age, y = lower), linetype = "dashed", color = "blue'
  geom_line(aes(x = age, y = upper), linetype = "dashed", color = "blue'
  scale_x_continuous(breaks = seq(1, 11, 1)) +
  labs(x = "Age", y = "Annual Surivial") +
  theme_xaringan()
```



Summary

- Could write a function to compute a delta method standard error in R by hand
- Can use the functions from `purrr` to apply a function easily to multiple inputs

Delta method in R via msm

msm R package

Package overview

- Multi-State Markov models
- From the online documentation:

> R package for continuous-time multi-state modeling of panel data
>
> - From the vignette:
>
> The multi-state Markov model is a useful way of describing a process in which an individual moves through a series of states in continuous time. The msm package for R allows a general multi-state model to be fitted to longitudinal data.
>
> - Provides an easier way to apply the delta method:
>   - Includes a `deltamethod` function
>   - Prevents the computation of partial derivatives (Yay! 😄)

`deltamethod` function from `msm`

Inputs:

- `g` = a formula representing the function: $g(\cdot)$

> The variables must be labeled `x1`, `x2`,...
>
> For example, if
>
> $$g(\hat{\boldsymbol{\beta}}) = \frac{1}{\hat{\beta}_0 + \hat{\beta}_1}$$
>
> then type function as: `g = ~ 1 / (x1 + x2)`
>
> - `mean` = vector of estimated parameters: $\hat{\boldsymbol{\theta}}$

- `cov` = estimated variance-covariance matrix: $Cov\left(\hat{\boldsymbol{\theta}}\right)$
- `ses`:
  - If TRUE, returns the standard errors of $g(\cdot)$ (default).
  - If FALSE, returns the variance-covariance matrix of $g(\cdot)$.

Example 1

Simple linear regression:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1$$

Compute standard error for

$$\frac{1}{\hat{\beta}_0 + \hat{\beta}_1}$$

```
# Simple linear regression
set.seed(1000)
x1 <- 1:100; y <- rnorm(100, 4*x1, 5)
m1 <- lm(y ~ x1)
# Extract the model coefficients and variance-covariance matrix
bhat1 <- coef(m1)
vc1 <- vcov(m1)
# Estimate of (1 / (b0hat + b1hat))
1 / (bhat1[[1]] + bhat1[[2]])
```

```
## [1] 0.4226072
```

```
# Approximate standard error
msm::deltamethod(g = ~ 1 / (x1 + x2), mean = bhat1, cov = vc1)
```

```
## [1] 0.175727
```

Example 2

Simple linear regression:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta} x_2$$

Compute standard error for

$$\frac{1}{\hat{\beta}_0 + \hat{\beta}_1}$$

```
# Simple linear regression
set.seed(1000)
x1 <- 1:100; x2 <- runif(100); y <- rnorm(100, 4*x1, 5)
m2 <- lm(y ~ x1 + x2)
# Extract the model coefficients and variance-covariance matrix
bhat2 <- coef(m2)[1:2]
vc2 <- vcov(m2)[1:2,1:2]
# Estimate of (1 / (b0hat + b1hat))
1 / (bhat2[[1]] + bhat2[[2]])
```

```
## [1] 0.1930787
```

```
# Approximate standard error
msm::deltamethod(g = ~ 1 / (x1 + x2), mean = bhat2, cov = vc2)
```

```
## [1] 0.04898642
```

Try it out

For the mule deer example, compute the standard error for $\hat{S}^4$ when $\mathbf{age} = \mathbf{1}$.

$$\hat{S}^4 = \left(\frac{\exp\left(\hat{\beta}_0 + \hat{\beta}_1 \cdot \mathbf{age} + \hat{\beta}_2 \cdot \mathbf{age}^2\right)}{1 + \exp\left(\hat{\beta}_0 + \hat{\beta}_1 \cdot \mathbf{age} + \hat{\beta}_2 \cdot \mathbf{age}^2\right)} \right)^4$$

Bonus: How to compute the standard error for $\hat{S}^4$ for an arbitrary age?

Try it out: Solution

For the mule deer example, compute the standard error for $\hat{S}^4$ when $\textbf{age} = 1$.

$$\hat{S}^4 = \left(\frac{\exp\left(\hat{\beta}_0 + \hat{\beta}_1 \cdot \textbf{age} + \hat{\beta}_2 \cdot \textbf{age}^2\right)}{1 + \exp\left(\hat{\beta}_0 + \hat{\beta}_1 \cdot \textbf{age} + \hat{\beta}_2 \cdot \textbf{age}^2\right)} \right)^4$$

```
# Create the form of the formula to put in the deltamethod function
formula = "~ (exp(x1 + x2 + x3)) / (1 + exp(x1 + x2 + x3))^4"
# Apply the deltamethod function
se = msm::deltamethod(
  as.formula(formula),
  mean = deer_betas,
  cov = deer_vcov
)
se
```

```
## [1] 0.0002382685
```

Bonus: How to compute the standard error for $\hat{S}^4$ for an arbitrary age in R?

> See next slide

Arbitrary age

Function for applying the msm function deltamethod to a specified age:

```
apply_msm_deltamethod <- function(age, betas, vcov){
  # Create the form of the formula to put in the deltamethod function
  formula <-
    sprintf("~ (exp(x1 + (x2 * %f) + (x3 * %f)) /
            (1 + exp(x1 + (x2 * %f) + (x3 * %f))))^4",
            age, age^2, age, age^2)
  # Apply the deltamethod function
  se = msm::deltamethod(as.formula(formula), mean = betas, cov = vcov)
  # Return the se in a dataframe
  return(data.frame(age, se))
}
```

```
apply_msm_deltamethod(1, deer_betas, deer_vcov)
```

```
##   age          se
## 1   1 0.07213014
```

Multiple ages

We could also use `map_df` to apply the function to multiple ages

```
purrr::map_df(
  .x = 1:11,
  .f = apply_msm_deltamethod,
  betas = deer_betas,
  vcov = deer_vcov
)
```

```
##    age         se
## 1    1 0.07213014
## 2    2 0.04527659
## 3    3 0.04126256
## 4    4 0.04523933
## 5    5 0.05009983
## 6    6 0.05449863
## 7    7 0.06257862
## 8    8 0.09162806
## 9    9 0.15634621
## 10  10 0.21683612
## 11  11 0.17733989
```

Summary

- `deltamethod` function in `msm` R package allows application of delta method without computing derivatives
- Need to input a formula that uses `x1`, `x2`,... for the parameters $\hat{\boldsymbol{\theta}}$
- Only extract the parameter estimates and corresponding variance-covariance matrix cells needed to compute the transformed parameter

Additional resources

- A good reference on the delta method that also describes how to apply the delta method in R can be found on the IDRE website.
- Another delta method function (`delta.method`) from the alr3 R package

A more complicated example

Canadian goose data

- Canada goose banding and recovery data using Burnham joint live-dead mark recapture models in RMark

- Burnham model uses Seber parameterization

  - S = survival
  - F = fidelity
  - r = dead recovery rate
  - p = live recapture rate

- Interested in point estimate and 95% confidence intervals for Brownie parameterization of dead recovery rate f:

$$f = r(1 - S)$$

- Not a way to change parameterization in RMark for Burnham models or a previously derived estimator for SE of f

- Let's use the delta method!

More details on the data

Three predictor variables:

- Age (3 levels):
  - Juvenile: 0 yrs old, binned as (0, 0.5]
  - Sub adult: 1 or 2 yrs old, binned as (0.5, 2.5]
  - Adult: 3 yrs or older, binned as (2.5, 23]
- Site (2 levels):
  - Rural
  - Urban
- Time (21 levels):
  - Year as a factor from 1999-2019

More details on the Rmark models

Mark recapture models:

- S, F, r, and p are all modeled using logistic regressions with formulas including age x site x time
- For example:

$$\log\left(\frac{r}{1-r}\right) \sim \textbf{age} \times \textbf{site} \times \textbf{time}$$

$$\log\left(\frac{S}{1-S}\right) \sim \textbf{age} \times \textbf{site} \times \textbf{time}$$

- Have beta estimates and the variance-covariance matrix for r and S

Load models and extract betas and variance-covariance matrix:

```
geese_models = readRDS("data/geese_model_results.rds")
geese_betas <- geese_models$results$beta
geese_vc <- geese_models$results$beta.vcv
```

Coefficients and variance covariance matrix

```
str(geese_betas)
```

```
## 'data.frame':    492 obs. of  4 variables:
##  $ estimate: num  1.017 -0.885 -0.165 0.282 0.54 ...
##  $ se      : num  0.805 7.243 0.822 3.616 1.073 ...
##  $ lcl     : num  -0.562 -15.082 -1.775 -6.805 -1.563 ...
##  $ ucl     : num  2.6 13.31 1.45 7.37 2.64 ...
```

```
head(geese_betas)
```

```
##                   estimate        se         lcl       ucl
## S:(Intercept)    1.0168546 0.8054464  -0.5618204  2.595530
## S:age(0.5,2.5] -0.8853253 7.2430156 -15.0816360 13.310986
## S:age(2.5,23]  -0.1645426 0.8218269  -1.7753233  1.446238
## S:siteUrban     0.2818604 3.6156143  -6.8047437  7.368465
## S:time2000      0.5401300 1.0729408  -1.5628339  2.643094
## S:time2001      1.4300495 0.8101876  -0.1579182  3.018017
```

```
str(geese_vc)
```

```
##  num [1:492, 1:492] 0.649 -4.491 -0.646 -1.883 -0.564 ...
```

Applying the delta method

Step 1: Need to determine $g(\boldsymbol{\beta})$ (that is, how is f related to the $\boldsymbol{\beta}$s)

This relationship will be passed into the `deltamethod` function.

Recall $f = r(1 - S)$ and let...

$$
\begin{aligned}
\eta_r(\boldsymbol{\beta}) &= \log\left(\frac{r}{1-r}\right) \\
&= \beta_{r,Intercept} \\
&\quad + \beta_{r,SubAdult} \times I[\text{age} = \text{sub adult}] \\
&\quad + \beta_{r,Adult} \times I[\text{age} = \text{adult}] \\
&\quad + \beta_{r,Urban} \times I[\text{site} = \text{urban}] \\
&\quad + \beta_{r,2000} \times I[\text{time} = 2000] \\
&\quad + \beta_{r,2001} \times I[\text{time} = 2001] \\
&\quad \vdots \\
&\quad + \beta_{r,Adult,Urban,2019} \\
&\qquad \times I[age = adult] \\
&\qquad \times I[site = urban] \\
&\qquad \times I[time = 2019]
\end{aligned}
$$

$$
\begin{aligned}
\eta_S(\boldsymbol{\beta}) &= \log\left(\frac{S}{1-S}\right) \\
&= \beta_{S,Intercept} \\
&\quad + \beta_{S,SubAdult} \times I[\text{age} = \text{sub adult}] \\
&\quad + \beta_{S,Adult} \times I[\text{age} = \text{adult}] \\
&\quad + \beta_{S,Urban} \times I[\text{site} = \text{urban}] \\
&\quad + \beta_{S,2000} \times I[\text{time} = 2000] \\
&\quad + \beta_{S,2001} \times I[\text{time} = 2001] \\
&\quad \vdots \\
&\quad + \beta_{S,Adult,Urban,2019} \\
&\qquad \times I[age = adult] \\
&\qquad \times I[site = urban] \\
&\qquad \times I[time = 2019]
\end{aligned}
$$

If we solve

$$\eta_r(\boldsymbol{\beta}) = \log\left(\frac{r}{1-r}\right) \quad \text{and} \quad \eta_S(\boldsymbol{\beta}) = \log\left(\frac{S}{1-S}\right)$$

for $\boldsymbol{r}$ and $\boldsymbol{S}$, we get:

$$r = \frac{e^{\eta_r(\boldsymbol{\beta})}}{1+e^{\eta_r(\boldsymbol{\beta})}} \quad \text{and} \quad S = \frac{e^{\eta_S(\boldsymbol{\beta})}}{1+e^{\eta_S(\boldsymbol{\beta})}}$$

Finally, we can relate $\boldsymbol{f}$ to the βs as:

$$\begin{aligned}
f &= r(1-S) \\
&= \frac{e^{\eta_r(\boldsymbol{\beta})}}{1+e^{\eta_r(\boldsymbol{\beta})}}\left(1-\frac{e^{\eta_S(\boldsymbol{\beta})}}{1+e^{\eta_S(\boldsymbol{\beta})}}\right) \\
&= \frac{e^{\eta_r(\boldsymbol{\beta})}}{1+e^{\eta_r(\boldsymbol{\beta})}}\left(\frac{1+e^{\eta_S(\boldsymbol{\beta})}-e^{\eta_S(\boldsymbol{\beta})}}{1+e^{\eta_S(\boldsymbol{\beta})}}\right) \\
&= \frac{e^{\eta_r(\boldsymbol{\beta})}}{(1+e^{\eta_r(\boldsymbol{\beta})})(1+e^{\eta_S(\boldsymbol{\beta})})} \\
&= g(\boldsymbol{\beta})
\end{aligned}$$

Applying the delta method

Step 2: Code $g(\boldsymbol{\beta})$ as a formula in R

Let's consider two examples:

(1) Age = Juvenile, Site = Rural, Time = 1999

> All of these categories are the reference categories in the model, which means they are contained in the intercept. As a result,
>
> $$g\left(\boldsymbol{\beta}\right) = f = \frac{e^{\eta_r(\boldsymbol{\beta})}}{(1 + e^{\eta_r(\boldsymbol{\beta})})(1 + e^{\eta_S(\boldsymbol{\beta})})} = \frac{e^{\beta_{r,0}}}{(1 + e^{\beta_{r,0}})(1 + e^{\beta_{S,0}})}.$$
>
> In R:

```
g_jr99 = "~ exp(x1) / ((1 + exp(x1)) * (1 + exp(x2)))"
```

(2) Age = Sub Adult, Site = Urban, Time = 2000

> None of these categories are reference categories, so this calculation will be a bit more complicated...

Again,

$$g(\boldsymbol{\beta}) = f = \frac{e^{\eta_r(\boldsymbol{\beta})}}{(1 + e^{\eta_r(\boldsymbol{\beta})})(1 + e^{\eta_S(\boldsymbol{\beta})})},$$

but in this situation,

$$\begin{aligned} \eta_r(\boldsymbol{\beta}) = \beta_{r,Intercept} \\ +\beta_{r,SubAdult} \\ +\beta_{r,Urban} \\ +\beta_{r,2000} \\ +\beta_{r,SubAdult,Urban} \\ +\beta_{r,SubAdult,2000} \\ +\beta_{r,Urban,2000} \\ +\beta_{r,SubAdult,Urban,2000} \end{aligned} \qquad \begin{aligned} \eta_S(\boldsymbol{\beta}) = \beta_{S,Intercept} \\ +\beta_{S,SubAdult} \\ +\beta_{S,Urban} \\ +\beta_{S,2000} \\ +\beta_{S,SubAdult,Urban} \\ +\beta_{S,SubAdult,2000} \\ +\beta_{S,Urban,2000} \\ +\beta_{S,SubAdult,Urban,2000}. \end{aligned}$$

In R:

```
g_su00_part1 = "(exp(x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8))"
g_su00_part2 = "(1 + exp(x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8))"
g_su00_part3 = "(1 + exp(x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16))'
g_su00 = paste0("~", g_su00_part1, "/",
                  "(", g_su00_part2, "*", g_su00_part3, ")")
```

Applying the delta method

Step 3: Obtain correct subset of βs and variance-covariance matrix

Source helpful function and cleaned up versions of betas and variance-covariance values:

```
source("code/geese_functions.R")
geese_betas_clean = read.csv("data/geese_betas_df.csv")
geese_vc_clean = read.csv("data/geese_vc_df.csv")
```

I wrote `get_betas_and_vc` to extract the betas and vc in a nice way for use in the `deltamethod` function (see the file geese_functions.R for more info)

```
res_jr99 <-
  get_betas_and_vc(
    age = "Juvenile",
    site = "Rural",
    time = 1999,
    betas_full =
      geese_betas_clean,
    vc_full =
      geese_vc_clean
  )
```

```
res_su00 <-
  get_betas_and_vc(
    age = "Sub Adult",
    site = "Urban",
    time = 2000,
    betas_full =
      geese_betas_clean,
    vc_full =
      geese_vc_clean
  )
```

```r
str(res_jr99)
```

```
## List of 3
##  $ case : chr [1:3] "Juvenile" "Rural" "1999"
##  $ betas:'data.frame':    2 obs. of  2 variables:
##   ..$ term: chr [1:2] "r:(Intercept)" "S:(Intercept)"
##   ..$ beta: num [1:2] -0.31 1.02
##  $ vc   : num [1:2, 1:2] 1.056 0.824 0.824 0.649
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "r:(Intercept)" "S:(Intercept)"
##   .. ..$ : chr [1:2] "r:(Intercept)" "S:(Intercept)"
```

```r
str(res_su00)
```

```
## List of 3
##  $ case : chr [1:3] "Sub Adult" "Urban" "2000"
##  $ betas:'data.frame':    16 obs. of  2 variables:
##   ..$ term: chr [1:16] "r:(Intercept)" "r:age(0.5,2.5]" "r:age(0.5,2.5]:sit
##   ..$ beta: num [1:16] -0.31 -4.6 9.35 -9.59 2.79 ...
##  $ vc   : num [1:16, 1:16] 1.06 11.57 -46.55 49.24 -13.18 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:16] "r:(Intercept)" "r:age(0.5,2.5]" "r:age(0.5,2.5]:site
##   .. ..$ : chr [1:16] "r:(Intercept)" "r:age(0.5,2.5]" "r:age(0.5,2.5]:site
```

Applying the delta method

Step 4: Input formula, betas, and variance-covariance into the **`deltamethod`** function

```
msm::deltamethod(
  g = as.formula(g_jr99),
  mean = res_jr99$betas$beta,
  cov = res_jr99$vc
)
```

```
## [1] 0.006699222
```

```
msm::deltamethod(
  g = as.formula(g_su00),
  mean = res_su00$betas$beta,
  cov = res_su00$vc
)
```

```
## [1] 0.02203537
```

Could go a step further and write a function

I wrote the function `compute_dmse` (see the file geese_functions.R for more info)

```
compute_dmse(
  age = "Juvenile",
  site = "Rural",
  time = 1999,
  betas_full = geese_betas_clean,
  vc_full = geese_vc_clean
)
```

```
##        age  site time          se
## 1 Juvenile Rural 1999 0.006699222
```

```
compute_dmse(
  age = "Sub Adult",
  site = "Urban",
  time = 2000,
  betas_full = geese_betas_clean,
  vc_full = geese_vc_clean
)
```

```
##         age  site time         se
## 1 Sub Adult Urban 2000 0.02203537
```

And even apply purrr

```r
# Create a data frame with all combinations of age, site, and year
var_values = expand.grid(
  age = c("Juvenile", "Sub Adult", "Adult"),
  site = c("Rural", "Urban"),
  time = 1999:2019
)
```

```r
# Apply compute_dmse to age, site, and time combinations
dmses <-
  purrr::pmap_df(
    .l = list(
      age = var_values$age,
      site = var_values$site,
      time = var_values$time
    ),
    .f = compute_dmse,
    betas_full = geese_betas_clean,
    vc_full = geese_vc_clean
  )
```

Results on next slide...

```r
# Print the results
dmses
```

```
##             age  site time          se
## 1    Juvenile Rural 1999 0.006699222
## 2   Sub Adult Rural 1999 0.105557655
## 3       Adult Rural 1999 0.010529005
## 4    Juvenile Urban 1999 0.044303939
## 5   Sub Adult Urban 1999 2.033184865
## 6       Adult Urban 1999 0.073883274
## 7    Juvenile Rural 2000 0.007520864
## 8   Sub Adult Rural 2000 0.028782852
## 9       Adult Rural 2000 0.008844291
## 10   Juvenile Urban 2000 0.037995040
## 11  Sub Adult Urban 2000 0.022035371
## 12      Adult Urban 2000 0.046603707
## 13   Juvenile Rural 2001 0.006023243
## 14  Sub Adult Rural 2001 0.017613253
## 15      Adult Rural 2001 0.006642993
## 16   Juvenile Urban 2001 0.017539830
## 17  Sub Adult Urban 2001 0.035064093
## 18      Adult Urban 2001 0.035557015
## 19   Juvenile Rural 2002 0.005919942
## 20  Sub Adult Rural 2002 0.006651031
## 21      Adult Rural 2002 0.005579023
## 22   Juvenile Urban 2002 0.022907490
## 23  Sub Adult Urban 2002 0.026073092
## 24      Adult Urban 2002 0.020918193
```

Summary

- For more complicated problems, be careful figuring out relationship between quantity of interest and the $\boldsymbol{\beta}$s
- Writing functions in R to apply the delta method helps to make process more efficient