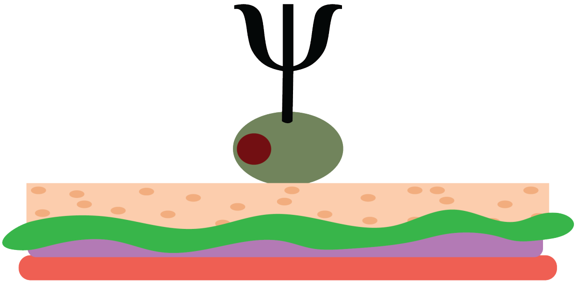
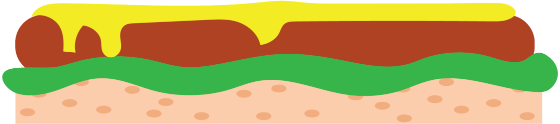


ABC's of Estimating Equations



Paul Zivich,[1] Rachael Ross,[2] Bonnie Shook-Sa[1]

[1]University of North Carolina, [2]Columbia University

Acknowledgements

Funding: K01AI177102 (PNZ), R01DA056407 (RKR), K01AI182506 (BES), R01AI157758 (PNZ, BES)

Disclaimer: views are ours and not those of NIH, DHHS, US government

pzivich@unc.edu   pzivich   pausalz@bsky.social

Run Code

github.com/pzivich/ABCs_of_M-estimation

- Open your preferred statistical software
- Open corresponding `mean.*` script
- Run the full script

```
Closed-form:  8.0
Root-finder:  8.0
95% CI: [ 0.8, 15.2]
```

Overview

A Terminological Note

Framework covered today goes by many names

- Estimating Equations
- M-estimation
- Z-estimation

May use terms interchangeably

Why Estimating Equations?

Learning estimating equations during my postdoc fundamentally changed how I think about and do epidemiology

- Approach problems from a different perspective

Made my work simpler by

- Making it easier to construct novel estimators
- Simplifying variance estimation[1]
- Being better equipped to read more theoretical papers
- Giving me a tool set to prove statistical properties

[1] I almost never use the bootstrap anymore!

Why Estimating Equations?

Metrika
https://doi.org/10.1007/s00184-024-00962-4

Variance estimation for average treatment effects estimated by g-computation

Stefan Nygaard Hansen[1] · Morten Overgaard[1]

Received: 3 February 2023 / Accepted: 8 March 2024

Why Estimating Equations?

Assume now that an estimator $\hat{\dot{\boldsymbol{\beta}}}_n(\mathbf{z})$ of $\dot{\boldsymbol{\beta}}(\mathbf{z})$ exists for all $\mathbf{z}$. The asymptotic covariance matrix of Theorem 2 may then be estimated by the following plug-in estimator

$$\hat{\boldsymbol{\Gamma}}_n^{\mathbf{a}} = \frac{1}{n}\sum_{i=1}^{n}\left\{\boldsymbol{\mu}(\hat{\boldsymbol{\beta}}_n; \mathbf{X}_i^{\mathbf{a}}) - \hat{\boldsymbol{\theta}}_n^{\mathbf{a}} + \left(\frac{1}{n}\sum_{j=1}^{n}\frac{\partial}{\partial\boldsymbol{\beta}}\boldsymbol{\mu}(\hat{\boldsymbol{\beta}}_n; \mathbf{X}_j^{\mathbf{a}})\right)\hat{\dot{\boldsymbol{\beta}}}_n(\mathbf{Z}_i)\right\}^{\otimes 2} \tag{8}$$

where $\mathbf{x}^{\otimes 2} = \mathbf{x}\mathbf{x}^T$ for a column vector $\mathbf{x}$.

Under some mild regularity conditions on the estimator $\hat{\dot{\boldsymbol{\beta}}}_n$, this plug-in estimator will be consistent for the asymptotic covariance matrix as the following result shows.

Theorem 3 *Make the assumptions of Theorem 2 and assume furthermore that $\hat{\dot{\boldsymbol{\beta}}}_n$ satisfies*

$$\|\hat{\dot{\boldsymbol{\beta}}}_n(\mathbf{z}) - \dot{\boldsymbol{\beta}}(\mathbf{z})\| \leq g_n \cdot f(\mathbf{z}) \tag{9}$$

for a sequence of random variables $g_n \xrightarrow{\mathrm{P}} 0$ and a measurable function f with $\mathrm{E}(f(\mathbf{Z})^2) < \infty$. Then $\hat{\boldsymbol{\Gamma}}_n^{\mathbf{a}} \xrightarrow{\mathrm{P}} \boldsymbol{\Gamma}^{\mathbf{a}}$.

Proof See the Appendix. □

Why Estimating Equations?

As an alternative to the two-step approach of this paper, one could consider formulating the two steps as two estimating equations and use (stacked) M-estimation. The sandwich variance estimator from the stacked M-estimation approach corresponds to the variance estimator of this paper. This M-estimation approach has been implemented in the Python library `delicatessen` as pointed out by a reviewer.

Estimating Equations Use-Cases

Causal inference

- Reifeis et al. (2020) 'Assessing exposure effects on gene expression' *Genetic Epidemiology*
- Tchetgen Tchetgen et al. (2024) 'Universal difference-in-differences for causal inference in epidemiology' *Epidemiology*
- Zivich et al. (2023) 'Introducing proximal causal inference for epidemiologists' *American Journal of Epidemiology*
- Zivich et al. (2024) 'Empirical sandwich variance estimator for iterated conditional expectation g-computation' *Statistics in Medicine*

Sensitivity analysis

- Cole et al. (2023) 'Higher-order evidence' *European Journal of Epidemiology*
- Cole et al. (2023) 'Sensitivity analyses for means or proportions with missing outcome data' *Epidemiology*

Measurement error

- Boe et al. (2024) 'Practical Considerations for Sandwich Variance Estimation in 2-Stage Regression Settings' *American Journal of Epidemiology*
- Ross et al. (2024)'Leveraging External Validation Data: The Challenges of Transporting Measurement Error Parameters' *Epidemiology*

Estimating Equations Use-Cases

Target trial emulation

- DeMonte et al. (2024) 'Assessing COVID-19 Vaccine Effectiveness in Observational Studies via Nested Trial Emulation' *arXiv:2403.18115*

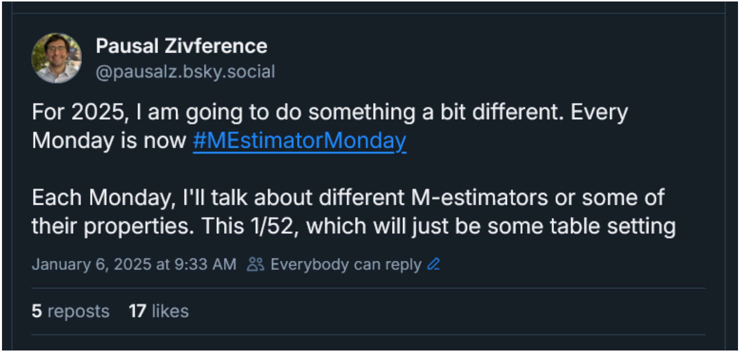
Generalizability / transportability

- Dahabreh, et al. (2020) 'Extending inferences from a randomized trial to a new target population' *Statistics in Medicine*
- Dahabreh, et al. (2023) 'Sensitivity analysis using bias functions for studies extending inferences from a randomized trial to a target population' *Statistics in Medicine*
- Robertson et al. (2024) 'Estimating subgroup effects in generalizability and transportability analyses' *American Journal of Epidemiology*
- Klose et al. (2025) 'Revisiting the Population Attributable Fraction' *Epidemiology*

Data fusion

- Cole et al. (2023) 'Illustration of 2 fusion designs and estimators' *American Journal of Epidemiology*
- Shook-Sa et al. (2024) 'Fusing trial data for treatment comparisons: single versus multi-span bridging' *Statistics in Medicine*

Estimating Equations Use-Cases



Overview

Section 1: introduction

Break (15min)

Section 2: applied examples

Break (15min)

Section 3: in context

Overview

Section 1: introduction

Break (15min)

Section 2: applied examples

Break (15min)

Section 3: in context

Overview: Section 1

Review notation / definitions

Estimating equations by-hand

Estimating equations with a computer

Some statistical properties

Notation

Review notation and mathematical operations used

- If unfamiliar with something, don't worry!
- Operations will be
  - Contextualized in following sections
  - Mainly done by the computer
- Resource for you to return to later

What we need:

- Basics
- Matrix algebra
- Derivatives

Notation – Basics

O_i: observed data for unit i

- $O_i = (X_i, Y_i)$

$\sum_{i=1}^{n} i \; = \; 1 + 2 + ... + n$: cumulative sum

$\prod_{i=1}^{n} i \; = \; 1 \times 2 \times ... \times n$: cumulative product

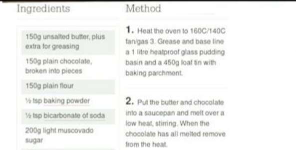
expit$(a) = 1/(1 + \exp(-a))$

$E[X]$: expected value function

Notation – Basics

estimand θ
(parameter of interest)

estimator $\hat{\theta}$



Ingredients

150g unsalted butter, plus extra for greasing
150g plain chocolate, broken into pieces
150g plain flour
½ tsp baking powder
½ tsp bicarbonate of soda
200g light muscovado sugar

Method

1. Heat the oven to 160C/140C fan/gas 3. Grease and base line a 1 litre heatproof glass pudding basin and a 450g loaf tin with baking parchment.

2. Put the butter and chocolate into a saucepan and melt over a low heat, stirring. When the chocolate has all melted remove from the heat.

estimate 0.5

 [2]

[2]Estimand also commonly denoted by θ_0 or θ^*

Notation – Vectors & Matrices

Vector: a list of numbers (or scalars)

$$A = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Matrix: a table of numbers

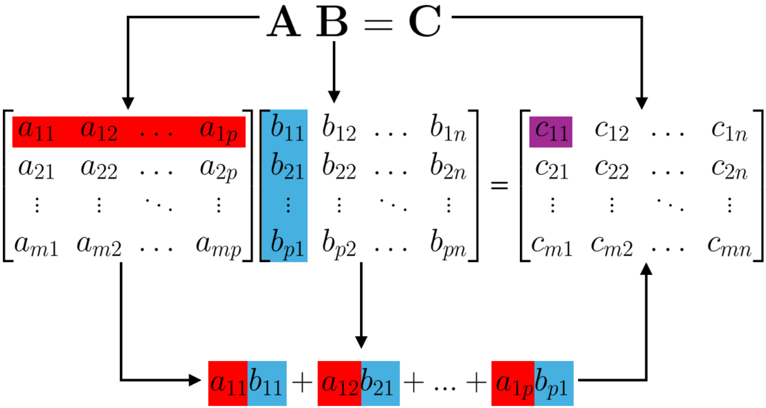
$$\mathbf{B} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Notation – Matrix Algebra

Transpose

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} \quad \mathbf{A}^T = \begin{bmatrix} a & c & e \\ b & d & f \end{bmatrix}$$

Dot product (matrix multiplication)



- Number of rows in first matrix must match columns in the second matrix

Notation – Matrix Algebra

Dot product (matrix multiplication)

$$\mathbf{A}\,\mathbf{B} = \mathbf{C}$$

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1p} \\ a_{21} & a_{22} & \dots & a_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mp} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{p1} & b_{p2} & \dots & b_{pn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mn} \end{bmatrix}$$

$$a_{21}b_{21} + a_{22}b_{21} + \ldots + a_{2p}b_{p1}$$

- Number of rows in first matrix must match columns in the second matrix

Notation – Matrix Algebra[3]

Inverse of 2×2 matrix

$$\mathbf{D} = \begin{bmatrix} w & x \\ y & z \end{bmatrix} \quad \mathbf{D}^{-1} = \frac{1}{w \; z \; - \; x \; y} \begin{bmatrix} z & -\,y \\ -\,x & w \end{bmatrix}$$

- Matrix must have same number of rows and columns

[3] I've never taken a linear algebra course, so don't worry if this matrix algebra isn't something you're familiar with

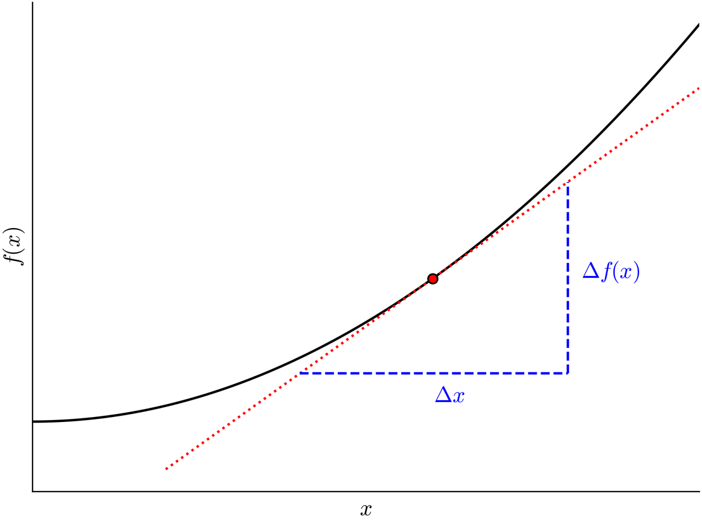
Derivatives – Basics

$$f'(x) = \frac{d}{dx} f(x)$$

Helpful to think of derivative as slope of tangent line at a point

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Derivatives – Basics



Derivatives – Generalizations

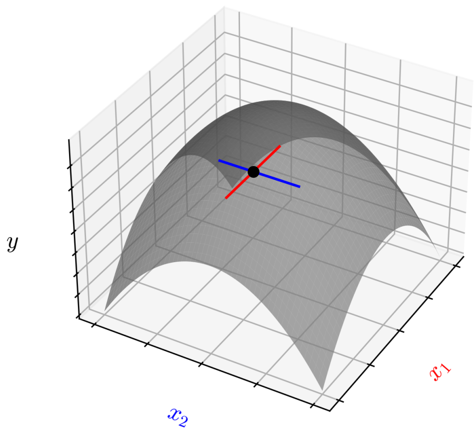
If $\boldsymbol{x} = (x_1, x_2, ..., x_m)$ and $f(\boldsymbol{x}) = y$, then the partial derivative is

$$\frac{\partial}{\partial x_1} f(\boldsymbol{x})$$

The gradient is

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(\boldsymbol{x}) \\ \frac{\partial}{\partial x_2} f(\boldsymbol{x}) \\ \vdots \\ \frac{\partial}{\partial x_m} f(\boldsymbol{x}) \end{bmatrix}$$

Derivatives – Generalizations



Derivatives – Generalizations

The Hessian is

$$\Delta H_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2}{\partial x_1 \partial x_1} f(\boldsymbol{x}) & \dots & \frac{\partial^2}{\partial x_1 \partial x_m} f(\boldsymbol{x}) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial x_m \partial x_1} f(\boldsymbol{x}) & \dots & \frac{\partial^2}{\partial x_m \partial x_m} f(\boldsymbol{x}) \end{bmatrix}$$

- Jacobian (transpose gradient, ∇^T) of the gradient

Derivatives – Generalization

Function

$$f(x_1, x_2) = y$$

Gradient

$$\nabla f(x_1, x_2) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(x_1, x_2) \\ \frac{\partial}{\partial x_2} f(x_1, x_2) \end{bmatrix}$$

Hessian

$$\Delta H_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2}{\partial x_1 \partial x_1} f(x_1, x_2) & \frac{\partial^2}{\partial x_1 \partial x_2} f(x_1, x_2) \\ \frac{\partial^2}{\partial x_2 \partial x_1} f(x_1, x_2) & \frac{\partial^2}{\partial x_2 \partial x_2} f(x_1, x_2) \end{bmatrix}$$

Notation – Estimating Equations

Estimating *function*

$$\psi(O_i; \theta)$$

Estimating *equation*

$$\sum_{i=1}^{n} \psi(O_i; \theta)$$

Estimator

Our estimator, $\hat{\theta}$, is the solution to

k-dimensional estimating function

k-dimensional parameter

$$\sum_{i=1}^{n} \boxed{\psi\left(O_i , \hat{\theta} \right)} = 0$$

Observation i

root: where $f(x) = 0$

Example 0: the mean

Problem: Learn the Mean

Want to learn the population mean

- Estimand: $\mu = E[Y]$

Suppose we have the following observations to estimate μ

$$7, 1, 5, 3, 24$$

Usual method

$$\underbrace{\hat{\mu}}_{\text{Estimator for mean}} = \frac{1}{\underbrace{n}_{\text{Total number of units}}} \sum_{i=1}^{n} \underbrace{Y_i}_{\text{Observed value for unit } i}$$

Applying to data in example (estimate)

$$\frac{7+1+5+3+24}{5} = \frac{40}{5} = 8$$

but let's use estimating equations instead

An Algorithm for Estimating Equations

1. Determine estimating function
2. Find the roots of the estimating equations
3. Estimate variance via the sandwich

1. Determine Estimating Function

Goal: rewrite mean as a function that is equal to zero

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} Y_i \qquad \text{def'n}$$

$$\hat{\mu} \; n = \sum_{i=1}^{n} Y_i \qquad \text{multiply by } n$$

$$0 = \left(\sum_{i=1}^{n} Y_i \right) - \hat{\mu} \; n \qquad \text{subtract } \hat{\mu} n$$

$$0 = \left(\sum_{i=1}^{n} Y_i \right) - \left(\sum_{i=1}^{n} \hat{\mu} \right) \qquad \text{def'n of } \times$$

$$0 = \sum_{i=1}^{n} (Y_i - \hat{\mu}) \qquad \text{associativity}$$

1. Determine Estimating Function

This formula is the estimating equation of the mean

$$\sum_{i=1}^{n} \underbrace{\overbrace{\psi}^{\text{Estimating function}} (\underbrace{O_i}_{\text{Observation } i} , \overbrace{\hat{\theta}}^{\text{Parameter}})} = \sum_{i=1}^{n} (\underbrace{Y_i}_{\text{Observation } i} - \overbrace{\hat{\mu}}^{\text{Parameter}}) = 0$$

2. Root-finding

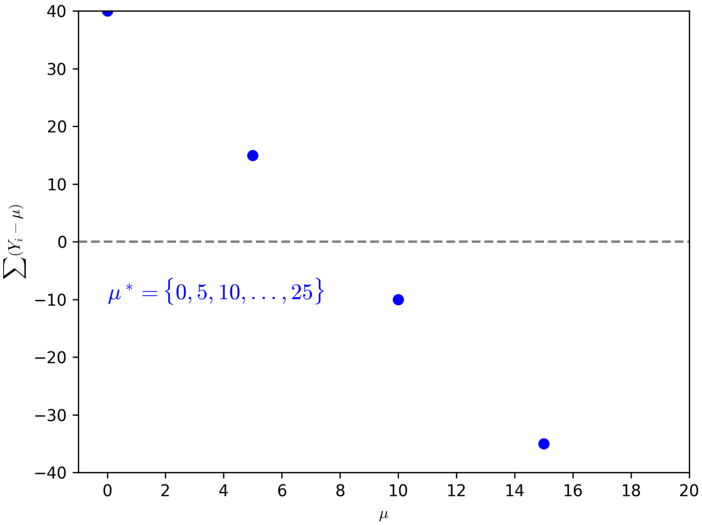
How can we find $\hat{\mu}$?

- Ignore the closed-form solution for the time

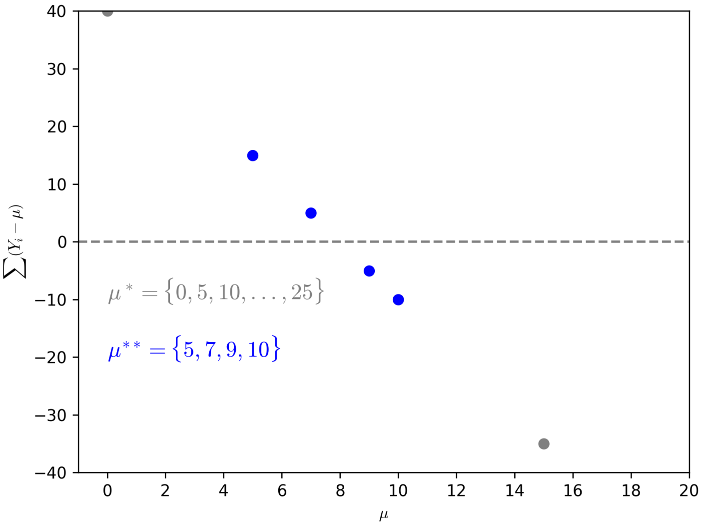
Broadly

1. Take some guesses at $\hat{\mu}$, denoted as $\hat{\mu}^*$
2. Compute $\sum_{i=1}^{n} \psi(O_i; \hat{\mu}^*)$
3. Find the guesses that are close to zero
4. Generate some new guesses, $\hat{\mu}^{**}$
5. Repeat 2-4 until we find $\hat{\mu}$

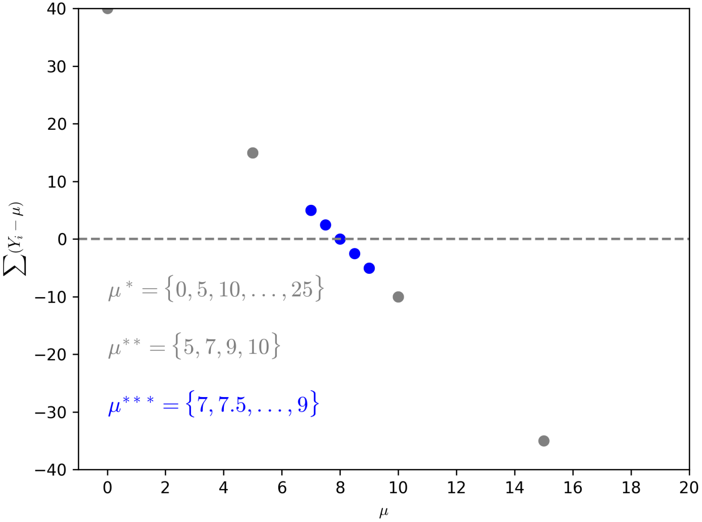
2. Root-finding



2. Root-finding



2. Root-finding



3. Variance

Closed-form estimator[4]

$$\widehat{Var}(\hat{\mu}) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{\mu})^2$$

but let's rely on estimating equations instead

[4]Note: n is often replaced by $n-1$ in practice, which can lead to differences for small sample sizes

3. Sandwich Variance Estimator

Sandwich variance

Filling (meat) matrix

$$\underbrace{V(\hat{\theta})} = \underbrace{B(\hat{\theta})^{-1}} \; \underbrace{F(\hat{\theta})} \left(\underbrace{B(\hat{\theta})^{-1}} \right)^T$$

(inverse of) Bread matrix

3. Sandwich Variance Estimator

Bread matrix

$$B(\hat{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \left[- \psi'(O_i, \hat{\theta}) \right]$$

Partial derivatives (Jacobian)

Filling matrix

$$F(\hat{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \left[\psi(O_i, \hat{\theta}) \; \psi(O_i, \hat{\theta})^T \right]$$

Dot product of estimating functions

Baking the Bread: By-Hand

Need the derivative of $\psi(O_i;\mu)$

$$\begin{aligned}
\psi'(O_i;\hat\mu) =& \frac{\partial}{\partial\hat\mu}\psi(O_i;\hat\mu) && \text{def'n} \\
=& \frac{\partial}{\partial\hat\mu}(Y_i-\hat\mu) && \text{def'n of estimating function} \\
=& -1 && \text{derivative rules}
\end{aligned}$$

Therefore

$$\boxed{\frac{1}{n}\sum_{i=1}^{n}\left[-\psi'(O_i,\hat\theta)\right]} = \frac{1}{n}\sum_{i=1}^{n}\left[-\boxed{-1}\right] = 1$$

Definition of Bread

From derivative above

Cooking the Filling: By-Hand

Definition of Filling

$$\boxed{\frac{1}{n}\sum_{i=1}^{n}\left[\psi(O_i,\hat{\theta})\psi(O_i,\hat{\theta})^T\right]} = \frac{1}{n}\sum_{i=1}^{n}\left[\boxed{(Y_i-\hat{\mu})(Y_i-\hat{\mu})}\right]$$

Plugging in estimating function

Therefore

$$\frac{1}{5}\sum_{i=1}^{5}\left[(Y_i-8)^2\right] = 68$$

Assembling the Sandwich: By-Hand

$$\underbrace{V(\hat{\mu})}_{\text{Sandwich variance}} = \underbrace{1^{-1}}_{\text{Bread}} \; \underbrace{68}_{\text{Filling}} \left(\underbrace{1^{-1}}_{\text{Bread}} \right)^{T} = 68$$

Wald-type confidence intervals

$$\hat{\mu} \pm z_{\alpha} \sqrt{\frac{V(\hat{\mu})}{n}} = 8 \pm 1.96 \sqrt{\frac{68}{5}} = (0.8, 15.2)$$

Computation of Estimating Equations

Computation of Estimating Equations

Solved estimating equation by-hand

- By-hand is not needed

Consider how estimating equations can be implemented algorithmically

- Root-finding
- Approximation of derivatives
- Matrix algebra

Follow along in `mean.R`, `mean.sas`, or `mean.py`

- Start of code inputs data and sets up estimating equations

Root-Finding – Algorithms

Performed a by-hand search for $\hat{\mu}$

- Similar to the *bisection method*

Variety of multidimensional root-finding algorithms exist[5]

- Secant method (quasi-Newton)
- Levenberg-Marquardt
- Powell hybrid method

[5] I've found Levenberg-Marquardt to be reliable for most problems

Root-Finding – Code

Under `Root-finding` see implementation

- SAS – `nlplm`
- R – `rootSolve::multiroot`
- Python – `scipy.optimize.root`

Derivatives – Back to the Definition

Derivative of function

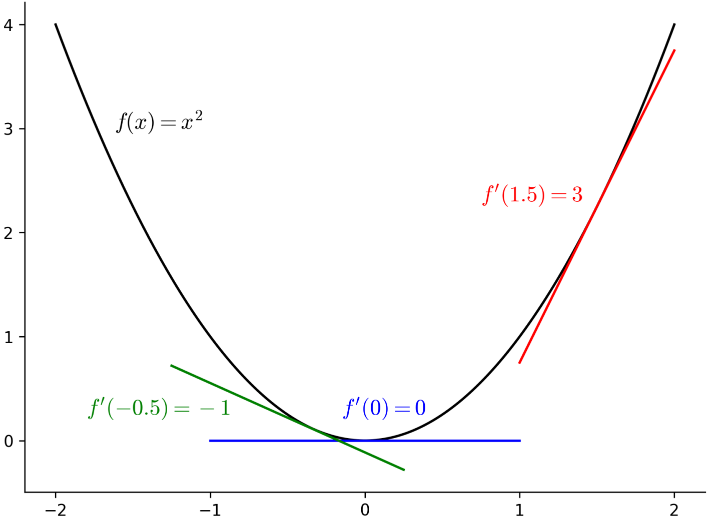
Change in output (rise)

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Behavior as h becomes small

Divided change in input (run)

Derivatives – Intuition



Central Difference Method[6]

$$\underbrace{\tilde{f}'(x)}_{\text{Approximation}} = \frac{f(\overbrace{x+a}^{\text{Slightly above } x}) - f(\overbrace{x-a}^{\text{Slightly below } x})}{2a}$$

Here a is a small value (e.g., 1×10^{-9})

[6] Automatic differentiation, which computes exact derivative, could be used instead. But this is not available in all software and is not straightforward to implement by-hand

Baking the Bread – Code

Under `Baking the bread` see implementation

- SAS – `nlpfdd`
- R – `numDeriv::jacobian`
- Python – `scipy.optimize.approx_fprime`

Cooking the Filling – Code

Under `Cooking the filling` see implementation

- Transpose
  - SAS – `'`
  - R – `base::t`
  - Python – `numpy.transpose`
- Dot product
  - SAS – `*`
  - R – `%*%`
  - Python – `numpy.dot`

Assembling the Sandwich – Code

Under `Assembling the sandwich` see implementation

- Inverse
  - SAS – `inv`
  - R – `base::solve`
  - Python – `numpy.linalg.inv`

Implications of our Algorithm

To evaluate estimating equations, we only need to provide

- Valid estimating functions
- Data

Everything else can be done by the computer

- Simplify complex analyses
- Open-source libraries
  - R: `geex`[7]
  - Python: `delicatessen`[8]

[7] Saul & Hudgens (2020) *Journal of Statistical Software*
[8] Zivich et al. (2022) *arXiv:2203.11300*

Extensions

But Why Estimating Equations?

All we've done is calculate the mean in a complicated way

So why bother with estimating equations?

- Flexibility of the framework

How Estimating Equations are extended

As will be seen in the next section

1. Stacking estimating functions
2. Automation of delta method

Stacking estimating functions

Often want to estimate more than 1 parameter

- Regression models
- Effect measure modification
- Inverse probability weighting

Stacking Estimating Functions

Stack estimating functions into a vector

$$\sum_{i=1}^{n} \begin{bmatrix} \psi_{\theta_1}(O_i; \hat{\theta}) \\ \psi_{\theta_2}(O_i; \hat{\theta}) \\ \vdots \\ \psi_{\theta_k}(O_i; \hat{\theta}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \mathbf{0}$$

- Easy to stack together
- Unlike maximizing a likelihood
  - Likelihood has a single value for individual contribution
  - More difficult to combine likelihood functions

Stacking Estimating Functions

Example

$$\sum_{i=1}^{n} \begin{bmatrix} \psi_{\theta_1}(O_i;\theta) \\ \psi_{\theta_2}(O_i;\theta) \end{bmatrix} = \sum_{i=1}^{n} \begin{bmatrix} Y_i - \theta_1 \\ (Y_i - \theta_1)^2 - \theta_2 \end{bmatrix} = \mathbf{0}$$

- Allow parameter to depend on others
- Concept explored further in applications

Delta Method

Theorem: smooth function of an asymptotically normal estimator is also asymptotically normal[9]

Application:

Transformation of α Covariance of α

$$Var\left\{\boxed{g(\alpha)}\right\} \approx g'(\alpha)\ \Sigma_\alpha\ g'(\alpha)$$

Derivative of transformation

[9] Boos & Stefanski *Essential Statistical Inference* pg. 237-240

Delta Method

Many variance formulas you know are Delta method results

- $Var(RD)$, $Var(\log(RR))$, $Var(\log(OR))$
- Formulas follow from Delta method argument
- Don't need to manually solve due to known formulas
  - Not always the case

Delta Method with the Sandwich

The estimating function for the transformed parameter, θ_t is

$$\psi_{g(\theta)}(O_i; \theta, \theta_t) = g(\theta) - \theta_t$$

- Estimating function does not depend on data

Therefore, the stacked estimating equations are

$$\sum_{i=1}^{n} \begin{bmatrix} \psi^*(O_i; \theta) \\ \psi_{g(\theta)}(O_i; \theta, \theta_t) \end{bmatrix} = 0$$

Delta Method with the Sandwich

Following some derivatives and matrix algebra

$$V(\theta, \theta_t) = \begin{bmatrix} V^*(\theta) & g'(\theta)V^*(\theta) \\ V^*(\theta)g'(\theta)^T & g'(\theta)V^*(\theta)g'(\theta) \end{bmatrix}$$

where

$$V(\theta_t) = \underbrace{g'(\theta)}_{\text{Derivative of transformation}} \overbrace{V^*(\theta)}^{\text{Sandwich covariance for } \theta} \underbrace{g'(\theta)}_{\text{Derivative of transformation}}$$

Automate the Delta method!

Robust Variance

To close this section, let's discuss the robust variance

- The sandwich variance is also known as the 'robust' variance
- 'Robust' designates that the variance estimator is not sensitive to violations of *certain* assumptions[10]
    - Variance estimator is consistent when parametric model is wrong
    - However this has some difficulties
- Relates back to Maximum Likelihood Estimation
    - The variance can be estimated two ways

[10]See Mansournia et al. (2021) *International Journal of Epidemiology* for further details

Robust Variance

Variance estimators

1. Inverse Hessian of the log-likelihood
   - Equivalent to $B(\theta)^{-1}$

2. Residuals of the score function
   - Equivalent to $F(\theta)^{-1}$

- When the model is correctly specified
  - These variance estimators asymptotically equivalent
  - $B(\theta) = F(\theta)$

Robust Variance

When the model is not correctly specified

- $B(\theta) \neq F(\theta)$
- By combining, sandwich is robust to assumptions
  - Variance estimator is consistent even if model is wrong

- Example: log-Poisson model to estimate the risk ratio
  - Here, estimated variance is too large

Warning[11]

- Does not correct for bias in parameter estimates

[11] See Freedman DA *Am Stat* 2006 for details

Overview

Section 1: introduction

Break (15min)

Section 2: applied examples

Break (15min)

Section 3: in context