Introduction

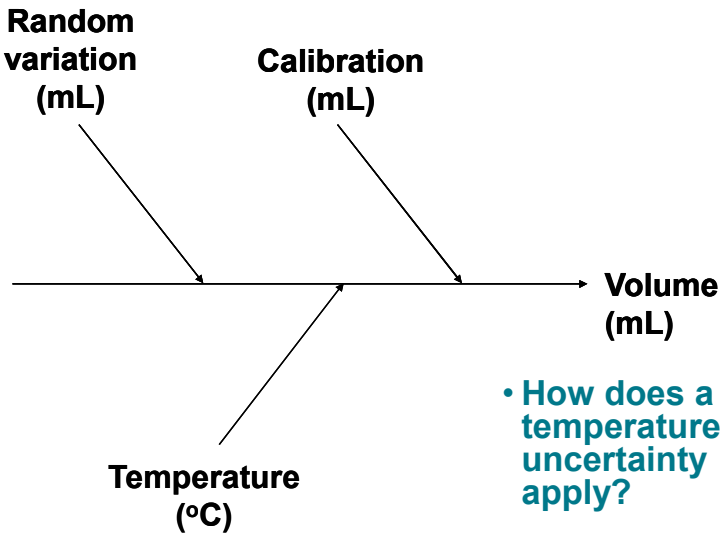
- Introduction to numerical methods for uncertainty evaluation
- Kragten's method
- Measurement uncertainty in R
  - The metRology package

A volumetric example

- Dispense 100ml
- from a Calibrated volumetric flask (U = 0.2 ml, k=2)
- allowing for random filling effects (s = 0.1 ml)
- at a laboratory temperature 20 ± 2 °C

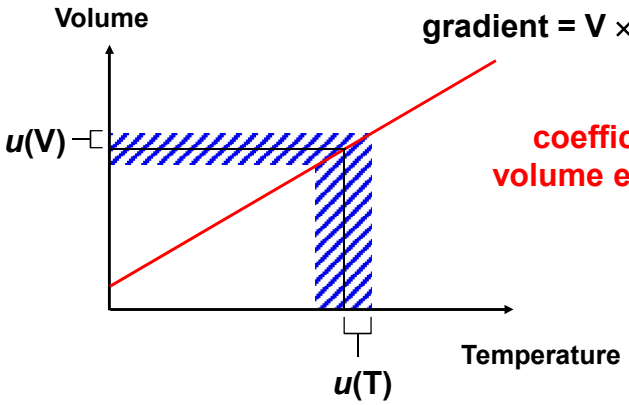
- Estimate the uncertainty in dispensed volume at 20 °C

Example: The effect of temperature on volume



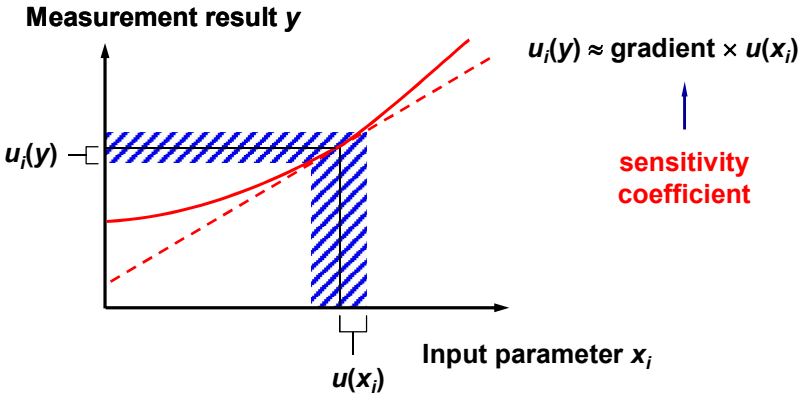
- **How does a temperature uncertainty apply?**

Example: The effect of temperature on volume



$u(V) = \text{gradient} \times u(T)$

Uncertainty propagation



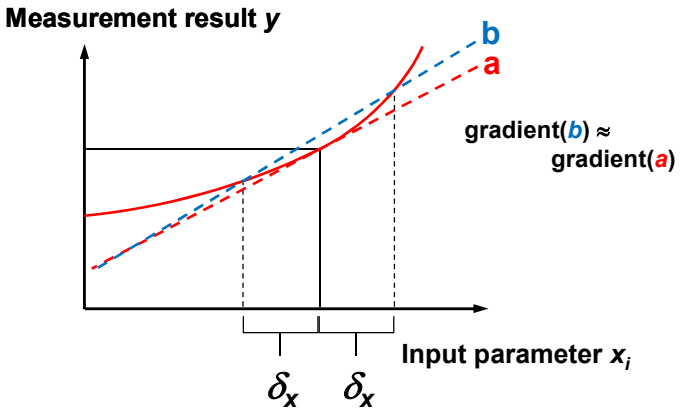
Mathematical form of uncertainty

- x_i parameter affecting analytical result y
- $u(x_i)$ uncertainty in x_i
- $u_i(y)$ uncertainty in y due to uncertainty in x_i

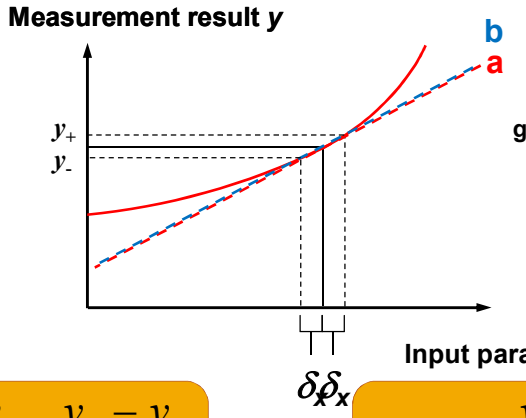
$$u_i(y) = \sqrt{\sum_i \left(\frac{\partial y}{\partial x_i}\right)^2 u(x_i)^2}$$

sensitivity coefficient

Finite difference method



Finite difference method



$$\frac{\partial y}{\partial x_i} \approx \frac{y_+ - y_-}{2\delta_{x_i}}$$

$$u_i(y) \approx \frac{y_+ - y_-}{2\delta_{x_i}} u(x_i)$$

Compare finite difference with the GUM

GUM first order

Expression: a/(b - c)

Uncertainty budget:

	x	u	c	u.c
a	1	0.05	1	0.05
b	3	0.15	-1	-0.15
c	2	0.10	1	0.10

y: 1
u(y): 0.1870829

Finite Difference

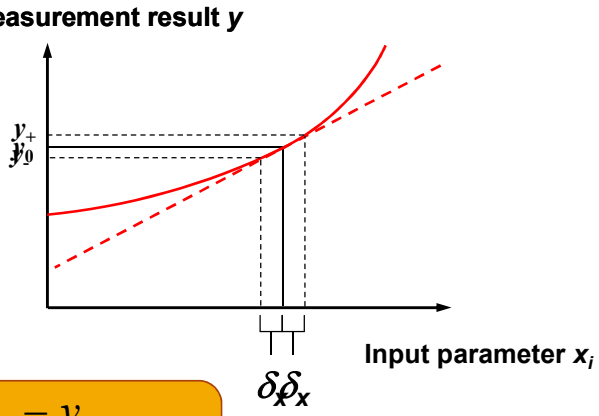
Expression: a/(b - c)

Uncertainty budget:

	x	u	c	u.c
a	1	0.05	1.000000	0.0500000
b	3	0.15	-1.000002	-0.1500003
c	2	0.10	1.000001	0.1000001

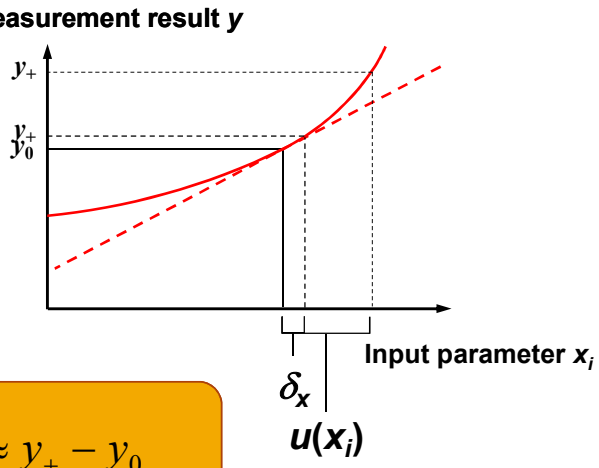
y: 1
u(y): 0.1870832

Kragten's method



$$u_i(y) \approx \frac{y_+ - y_0}{\delta_{x_i}} u(x_i)$$

Kragten's method



$$u_i(y) \approx y_+ - y_0$$

Compare Kragten with FD

Finite Difference

Expression: a/(b - c)

Uncertainty budget:

	x	u	c	u.c
a	1	0.05	1.000000	0.0500000
b	3	0.15	-1.000002	-0.1500003
c	2	0.10	1.000001	0.1000001

y: 1

u(y): 0.1870832

Kragten

Expression: a/(b - c)

Uncertainty budget:

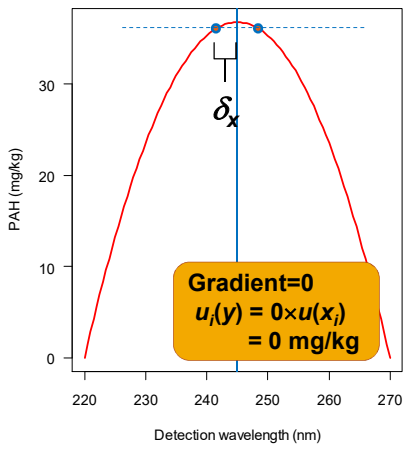
	x	u	c	u.c
a	1	0.05	1.0000	0.05000
b	3	0.15	-0.8695	-0.13043
c	2	0.10	1.1111	0.11111

y: 1

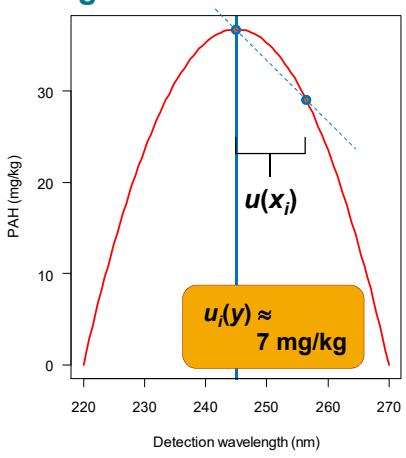
u(y): 0.1784906

Why use a 'less accurate' method?

Finite difference



Kragten



Finite difference methods compared

Finite difference 1st order

- Accurate gradient
- Faithfully reproduces 1st order GUM uncertainty
- Simple to calculate

- 1st order GUM is insufficient for highly non-linear cases
  - Needs 2nd and higher order

Kragten

- Exact only for linear examples
- Does not reproduce 1st order GUM
- Simple to calculate

- Usually adequate for mild nonlinearity
- May be better for highly non-linear cases

Both much simpler than manual differentiation

Measurement uncertainty in R: The metrology package

What is …

- R
  - "R is an integrated suite of software facilities for data manipulation, calculation and graphical display"
    - Free, open source package for statistical analysis and programming
    - Extensible via "packages"
- The metRology package
  - An R package for statistics applied to metrology
  - metrology: The science of measurement

Functionality: What does it do?

"metRology provides classes and calculation and plotting functions for metrology applications, including measurement uncertainty estimation and inter-laboratory metrology comparison studies."

https://r-forge.r-project.org/projects/metrology/

25 September 2017

- Measurement uncertainty estimation
- Support for interlaboratory studies

Uncertainty implementations in metRology

- **Direct combination of contributions**

$$u_i(y) = u(x_i)\frac{\partial y}{\partial x_i}$$

- **Algebraic differentiation of $f(x_1, \ldots)$**

- **Numerical differentiation**
  - Kragten's method

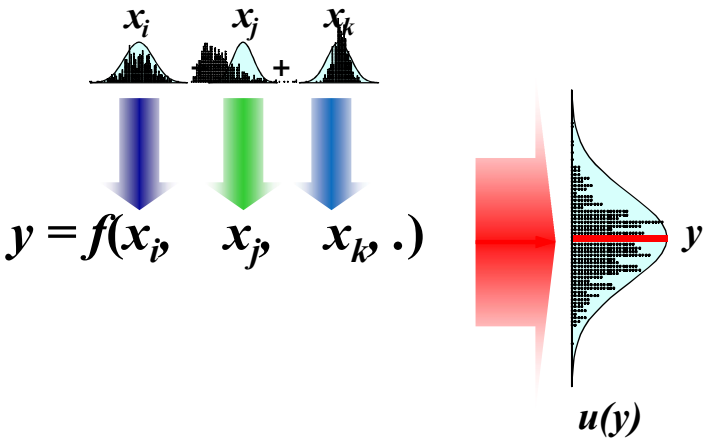
$$u(x_i)\partial y/\partial x_i \approx f(x_i + u(x_i)) - f(x_i)$$

  - Symmetric finite difference

$$\partial y/\partial x_i \approx [f(x_i + \delta x_i) - f(x_i - \delta x_i)]/2\delta x_i$$

Uncertainty implementations in metRology

2. Monte Carlo (GUM Supplement 1)



Examples

```
expr <- expression(a+b*2+c*3+d/2)
       # a=1(0.1), b=3(0.3), c=2(0.2), d=11(1.1)
u.expr<-uncert(expr, x, u, method="NUM")
```

```
#Compare with default:
uncert(u=c(0.1, 0.3, 0.2, 1.1), c=c(1.0, 2.0, 3.0, 0.5))

#... or with function method
 f <- function(a,b,c,d) a+b*2+c*3+d/2
 u.fun<-uncert(f, x, u, method="NUM")

#.. or with the formula method
 u.form<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM")
```

Examples (cont.)

```
Uncertainty evaluation
Call:
  uncert.expression(expr = expr, x = x, u = u, method =
 "NUM")
Expression: a + b * 2 + c * 3 + d/2
```

```
Evaluation method:  NUM
Uncertainty budget:
  x  u   c   u.c
a  1 0.1 1.0 0.10
b  3 0.3 2.0 0.60
c  2 0.2 3.0 0.60
d 11 1.1 0.5 0.55
```

```
   y:  18.5
u(y):  1.01612
```

A more interesting Monte carlo case

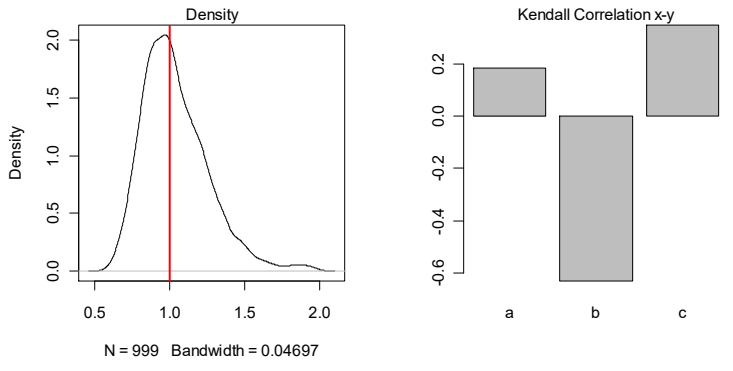
```
expr <- expression(a/(b-c))
x <- list(a=1, b=3, c=2)
Expression: a/(b - c)

   y:  1
u(y):  0.2187031

Monte Carlo evaluation using 999 replicates:
   y:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.6047  0.8845  1.0020  1.0410  1.1630  1.9540
```

A more interesting Monte carlo case

```
> par(mfrow=c(2,2))
> plot(u.invexpr, which=1:4, pch=20, method="k")
                # method="k" gives Kendall correlation
```



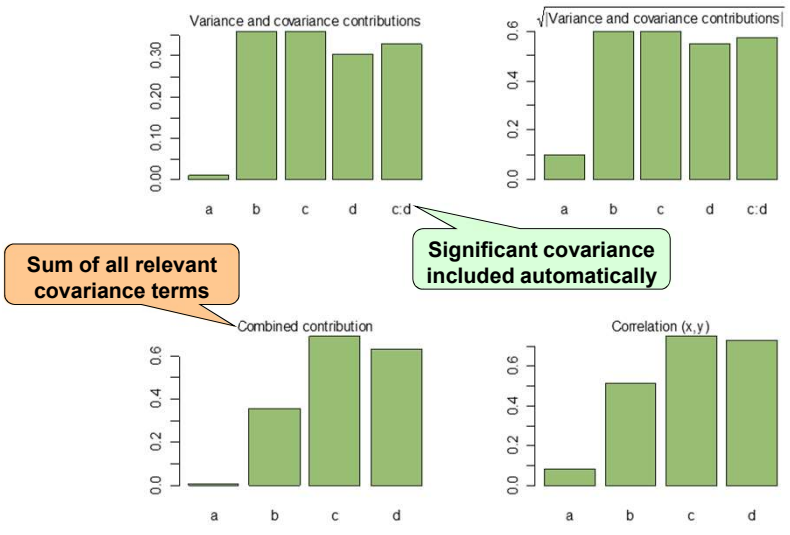
Other MU diagnostics and utilities

```
#An example with correlation
>  u.cor<-diag(1,4)
>  u.cor[3,4]<-u.cor[4,3]<-0.5
>  u.formc<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM",
+
>
#Which uncertainties matter most?
> par(mfrow=c(2,2)
> plot(u.formc)
```

Builds a correlation matrix

Plots for exploring uncertainty budgets

Other MU diagnostics and utilities: plot.uncert



Other MU diagnostics and utilities

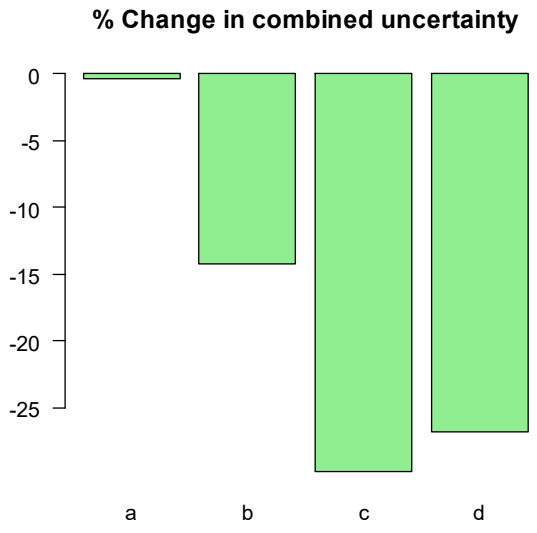
```
#An example with correlation
>  u.cor<-diag(1,4)
>  u.cor[3,4]<-u.cor[4,3]<-0.5
>  u.formc<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM",
+
>
```

```
#Which uncertainties matter most?
> par(mfrow=c(2,2)
> plot(u.formc)

#What happens if we reduce one uncertainty?
> barplot( drop1(u.formc) )
```

Drop each term successively

Other MU diagnostics and utilities: drop1



Other MU diagnostics and utilities

- update
  - Modifies uncertainty budgets (for example, changing an individual uncertainty or the method of evaluation)
- w.s, welch.satterthwaite
  - Welch-Satterthwaite effective degrees of freedom
- buildCov, buildCor
  - simplifies assembly of correlation or covariance matrices by taking a short list of labelled off-diagonal terms

Who owns metRology?

- metRology is an open source project
- No single ‘owner’
- Licence is GPL
  - Copyright is held by code contributor
  - Contribution is conditional on granting full permissions under the GPL:
    - right to distribute and modify under the same terms

Location: Where is metRology?

- CRAN:

```
install.packages("metRology")
```

- R-Forge: metRology
  https://r-forge.r-project.org/projects/metrology/

- Code access via SubVersion (svn)
  - Tortoise SVN for Windows

Acknowledgements

- NIST Statistical Engineering Division
  - Support and code contribution