# FLAMe case study

*Luke Loken*

*April 03, 2018*

## What do I do?

- Characterize within lake patterns of water chemistry
- Interpolate to generate spatially weighted distributions of water chemistry
- Automate these steps across multiple lakes and through time

## Code reflects my progression performing spatial analyses in R

- Old code was long scripts, loops, copy/paste
- New code using functions/apply statements, but I have yet to go back and tidy old code.

## Data example

- Methane measurements while boating around Little Satin Germain Lake.
- ~10,000 point observations distributed in grid like transects
- We can interpolate across the lake surface and estimate CH4 flux to the atmosphere for the entire lake?
- We also ask questions about the drivers and consequences of spatial heterogeneity within lakes

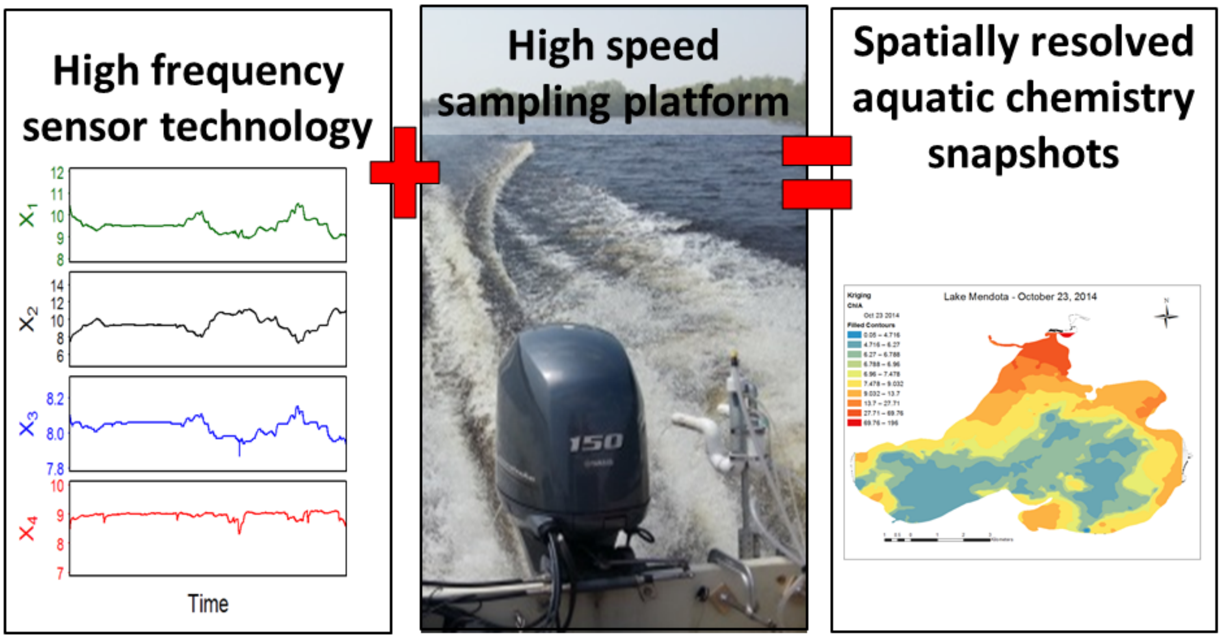
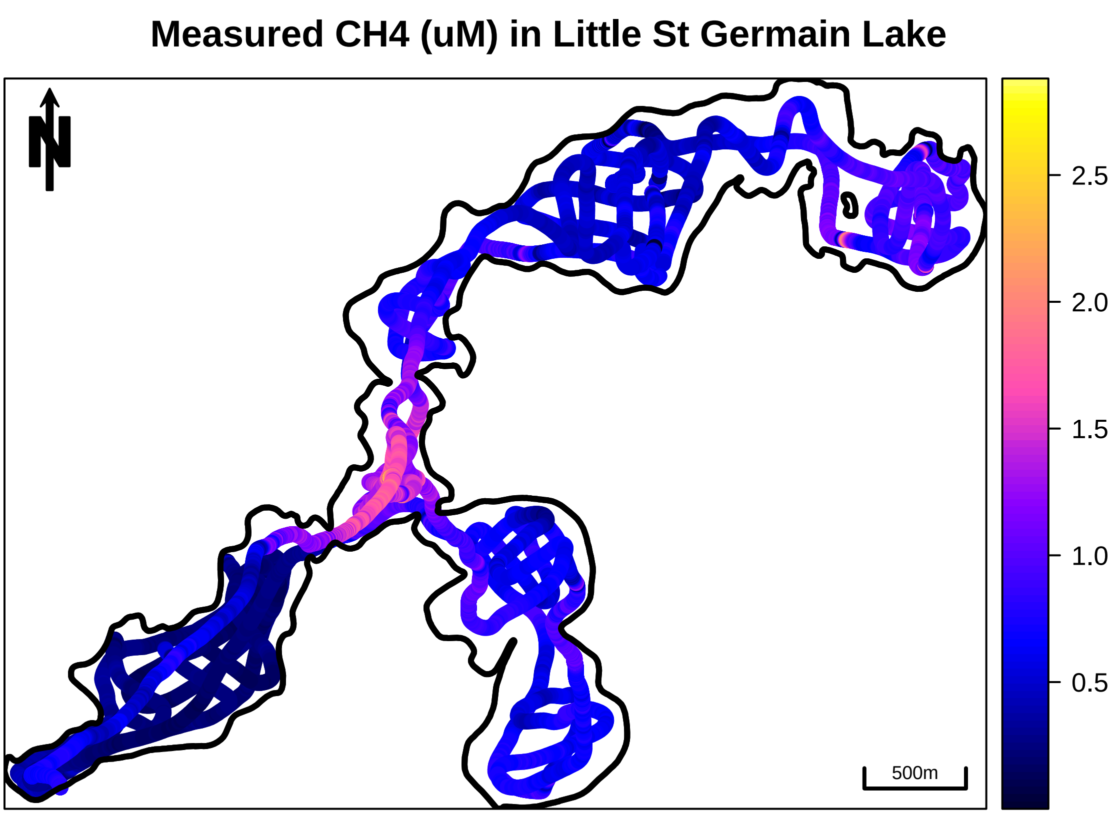
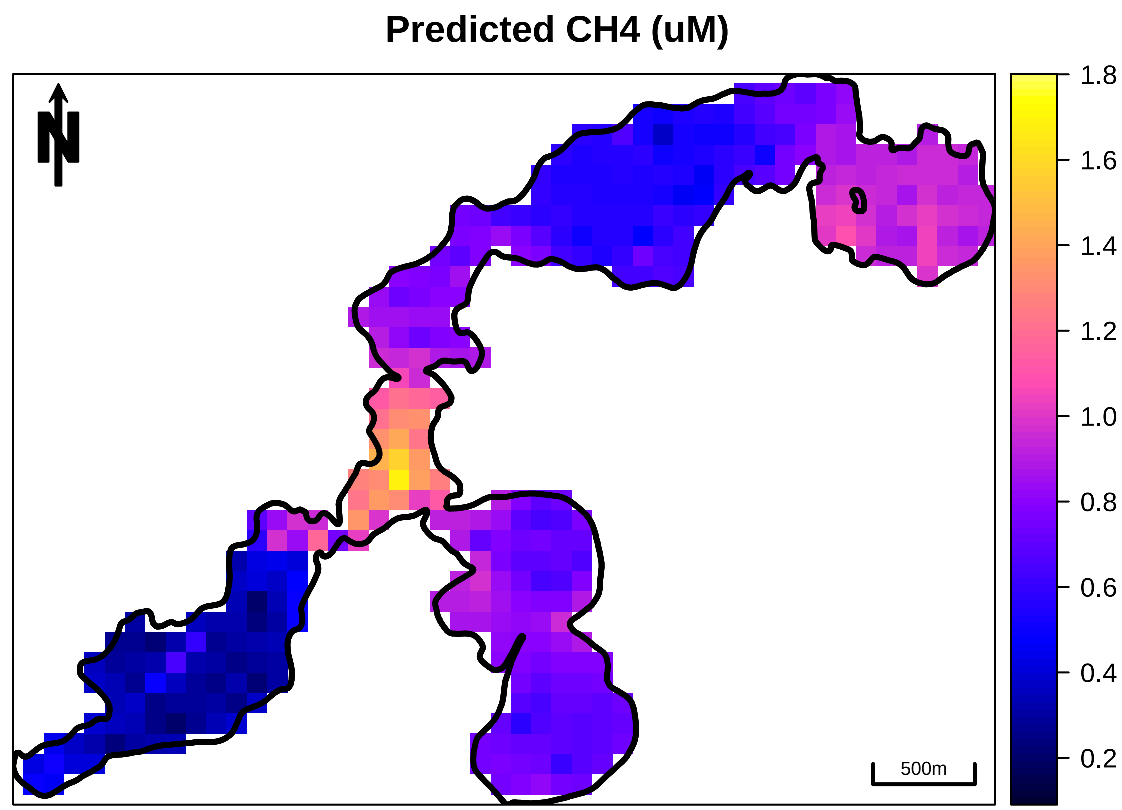


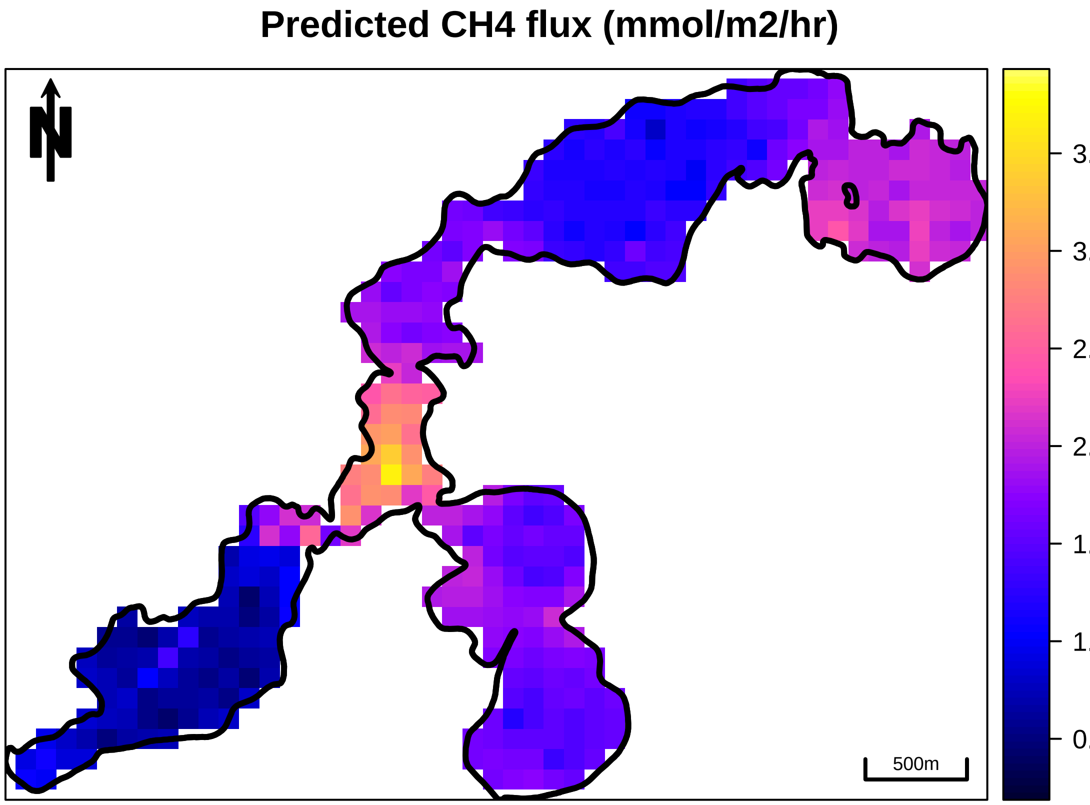

Figure 1: Crawford, Loken et al. 2015

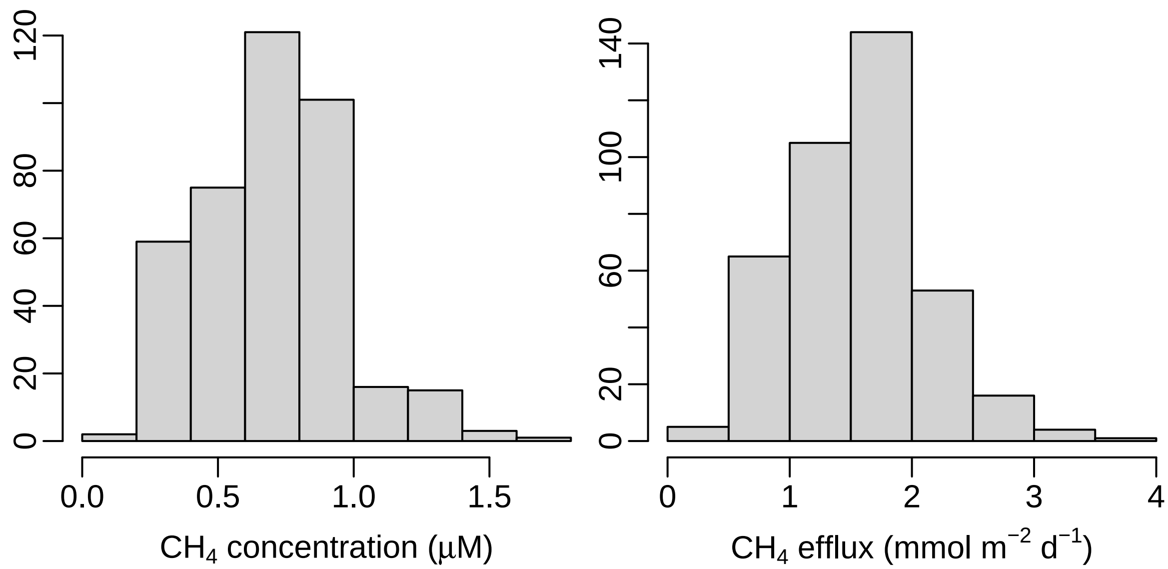


```
## [inverse distance weighted interpolation]
```




Convert Concentration to flux using a model of gas transfer velocity




Clearly spatial patterns exist for methane concentration/flux. Estimates from a single location do not represent the entire lake surface.

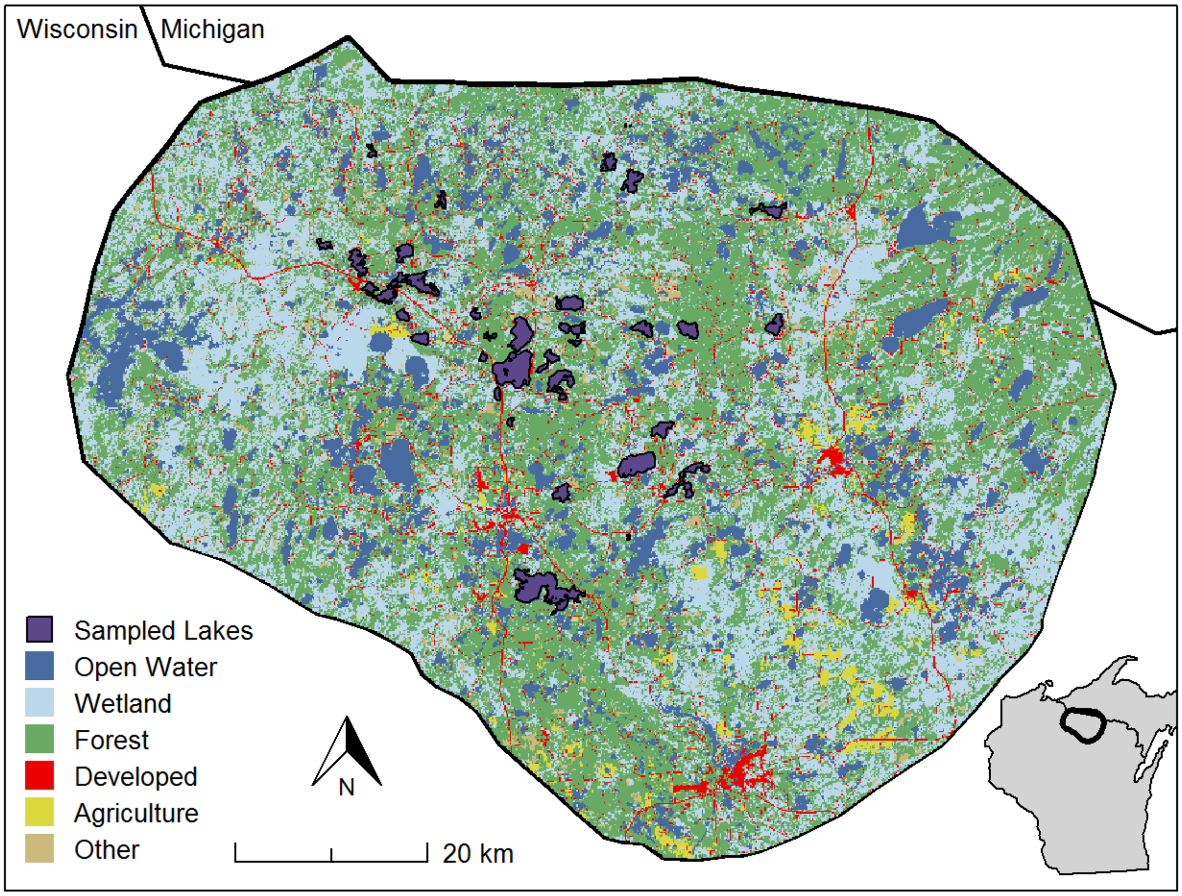



Figure 2: Northern Highland Lake District

## General questions to my research

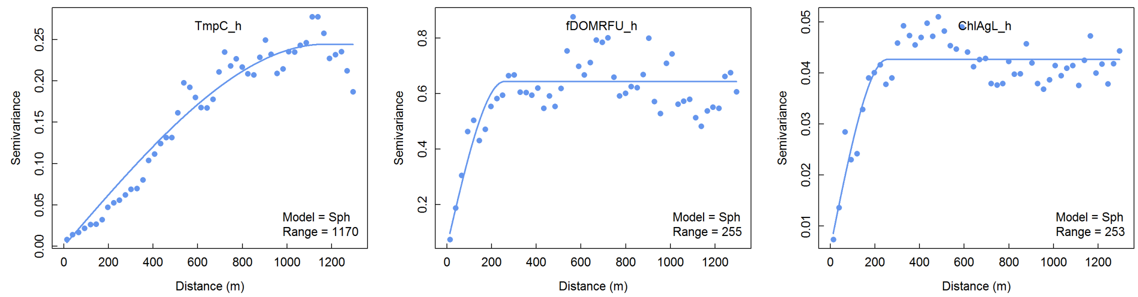
- How variable are individual lakes?
- How does variability vary among variables? (Temperature vs dissolved oxygen)
- How does variability vary among lakes? (Big lakes vs small lakes)
- What are the dominant scales of variation? (small patches vs large patches)

## Example 2 - Within lake spatial variability among lakes

- Sampled 40 lakes in Northern Wisconsin across lake productivity, size, and morphometry gradients
- For each lake, statistics describing spatial heterogeneity were calculated for all measurements (Temperature, dissolved oxygen, etc.)
  - standard deviation
  - quartile dispersion
  - coefficient of variation
  - skewness
- We also evaluated spatial structure using semivariance and correlogram models

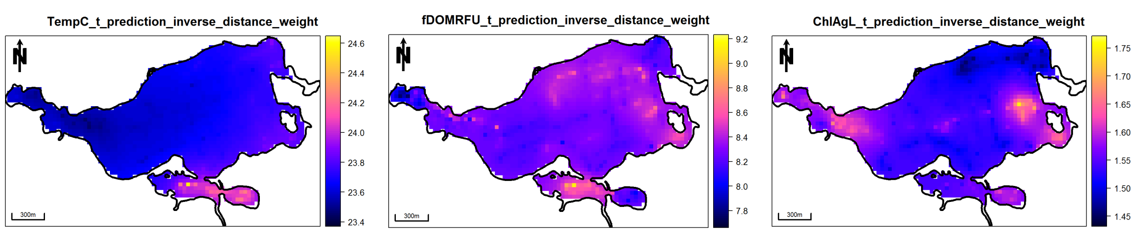
– semivariance range
– correlation at distance intervals

Semivariance models for 3 variables




Semivariance range is the distance at which observations are no longer spatially autocorrelated.

In this example, temperature (left panel) has a greater semivariance range (~1000 m), while fDOM (middle) and chlorophyll (right) have shorter ranges (~250 m). In this example, we can say that the dominant scale of variation for temperature occurs at a broader scale than the other variables.

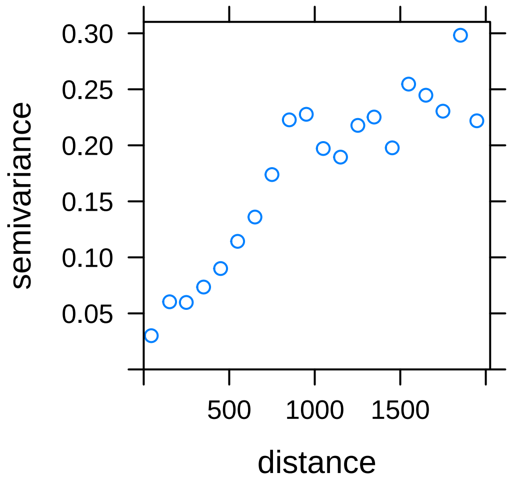



Looking at the maps, we see finer patches for fDOM and chlorophyll than temperature

## Semivariance modelling in R

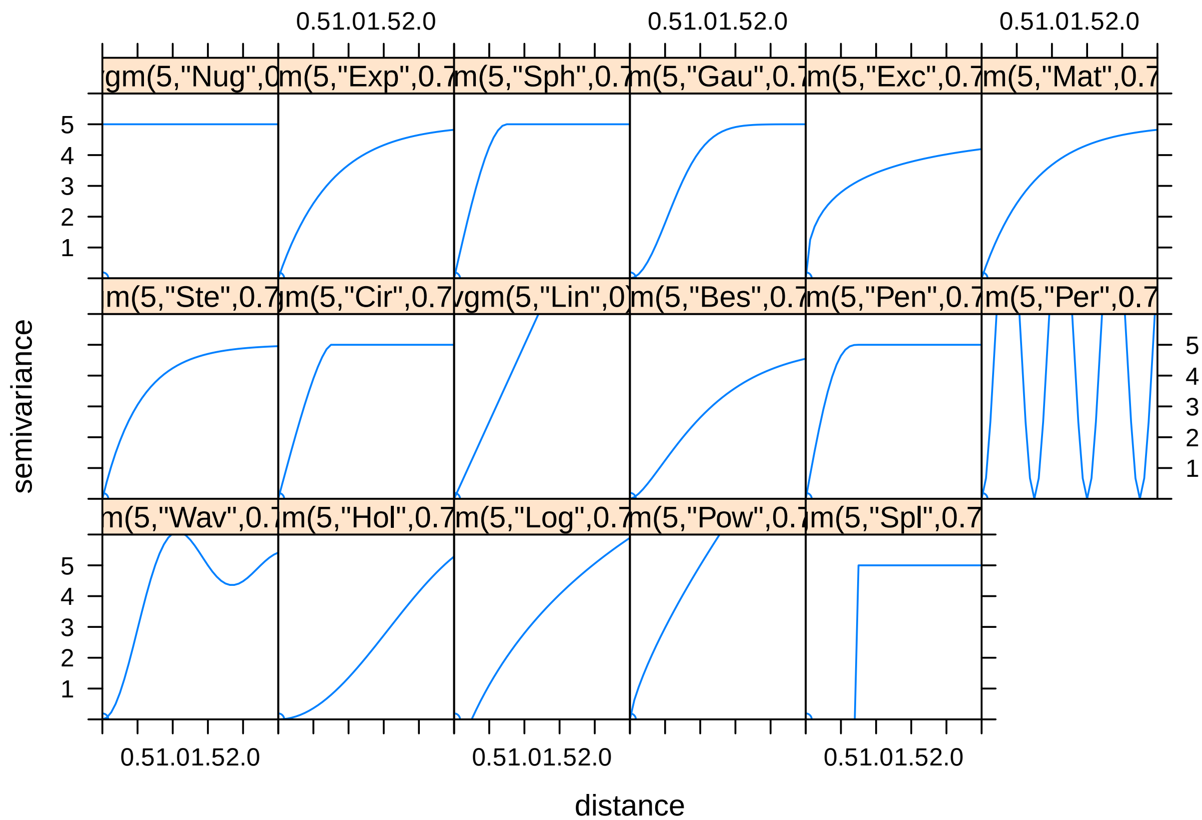
- Uses the functions `variogram()`, `vgm()`, `fit.variogram()` in the `gstat` package

```
library(gstat)

#Parameters for semivariogram calculation. Maximum distance (cutoff) and bin size (width)
cutoff=2000
width=100

# CH4 data are the 50th column in spatial points data frame 'LSG2'

# Calculate variogram
v = variogram(LSG2@data[,50]~1, LSG2, cutoff=cutoff, width=width)
plot(v)
```



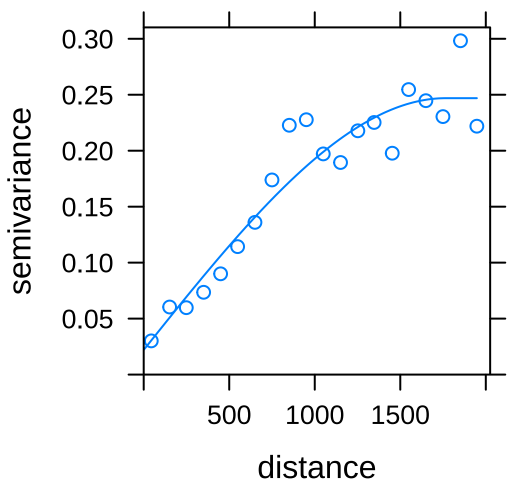

- Now we need to fit a model
- What kind of model should we use?

```r
#Look at a few types of semivariogram models
show.vgms(sill=5, range=0.75, ylim=c(0,6), xlim=c(0,2.5))
```



- Based on the variogram of the data, many choices are appropriate.
- Note that these models all have the same sill and range parameter, but show different plateau distances. In all of these models the `range==1`, however only in the Spherical and circular models does this equate to the distance at which the sill is reached. Other models (Exponential, Gaussian, etc.) continue increasing beyond this distance. Therefore be careful in interpreting the 'range' attribute that these models produce.

```
models<-c('Nug', 'Lin', 'Sph', 'Gau')

# Guess sill, range, and nugget
# These help the fit.variogram function figure out the best model
est_sill<-median(v$gamma)
est_range<-cutoff/4
est_nugget<-v$gamma[1]

#fit model to variogram
#fit.method=2 weights closer observations more than distant ones.
v.fit <- fit.variogram(v, vgm(est_sill=est_sill, models, est_range=est_range, nugget=est_nugget), fit.me

# Look at model fit
plot(v, v.fit)
```




```
v.fit
```

```
##   model      psill    range
## 1   Nug 0.02203928    0.000
## 2   Sph 0.22489710 1765.931
```

```
#output semivariogram parameters
SemiRange <- v.fit$range[2]
SemiSill <- v.fit$psill[2]
```

Looks like a spherical model was selected, which does a pretty good of fitting data. `fit.variogram` uses a weighted sum of squared errors for fitting models You can adjust the weighting using the `fit.method` attribute in the `vgm` function

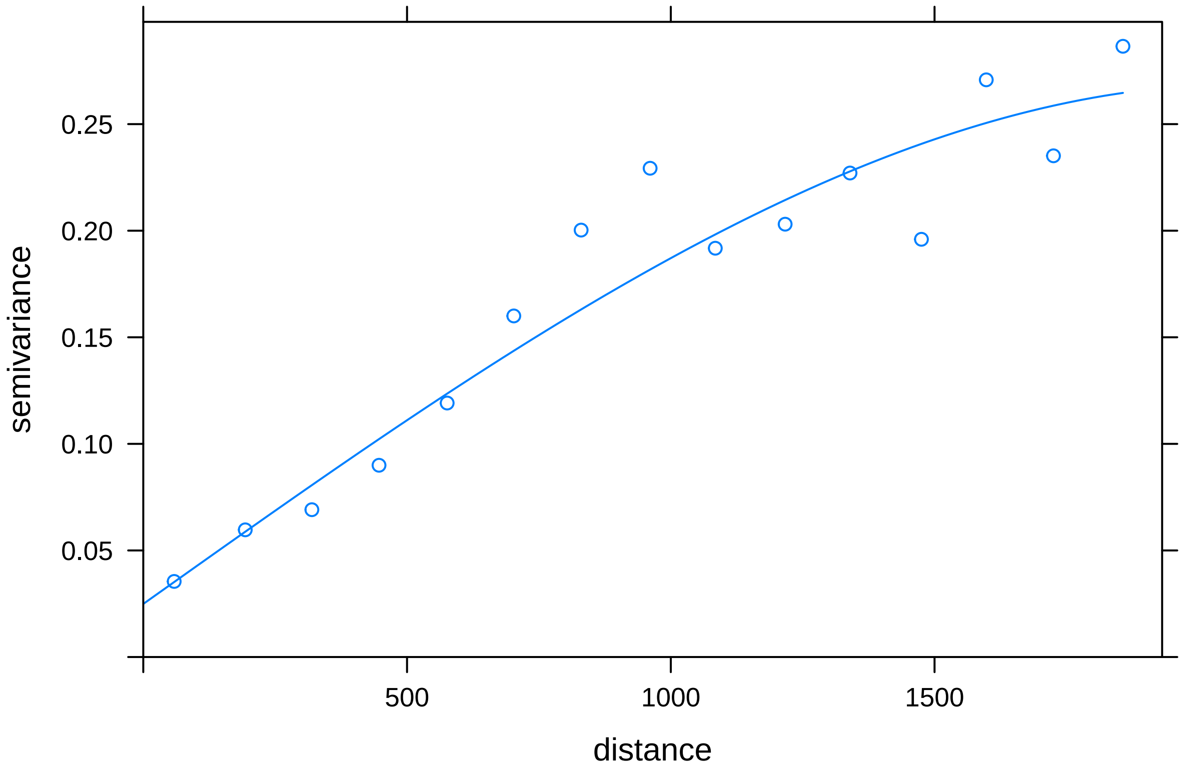
Or do this with a function and extract for multiple variables, days, and lakes

```r
# Function to fit variograms and extract range/model/sill
ExtractRangeSill <-function (SPdf, colname, models, plot, ...){

  #Subset data to remove NAs
  SPdf2<-SPdf[is.na(SPdf@data[,colname])==FALSE,]

  #Create variogram and estimate model parameters
  v <- variogram(SPdf2@data[,colname]~1, SPdf2, ...)
  est_sill<-median(v$gamma)
  est_range<-max(v$dist)/4
  est_nugget<-v$gamma[1]

  #Fit variogram model to variogram
  v.fit<-fit.variogram(v, vgm(est_sill=est_sill, models, est_range=est_range, nugget=est_nugget), ...)

  #output model type, sill, and range
  output<- data.frame(model=as.character(v.fit$model[nrow(v.fit)]), sill=v.fit$psill[nrow(v.fit)], range

  #Optional plotting
  if (missing(plot)){
  } else if (plot==T){
    print(plot(v, v.fit))
  } else {
  }

  return(output)

}

#Let's try it for one variable
head(LSG) #LSG is the spatialpointsdataframe
```

```
##                 ltime    Depth XCO2Dpp XCH4Dpp  TempC SPCuScm ChlARFU
## 1 2015-07-15 19:46:00 3.795174 92.4604 170.658 25.738    74.9    0.13
## 2 2015-07-15 19:46:01 3.866333 91.8642 170.928 25.732    75.0    0.13
## 3 2015-07-15 19:46:02 3.961216 91.7539 171.028 25.724    75.0    0.14
## 4 2015-07-15 19:46:03 3.913776 91.0657 171.618 25.715    75.0    0.14
## 5 2015-07-15 19:46:04 4.174694 91.3500 172.056 25.706    75.0    0.14
## 6 2015-07-15 19:46:05 3.961216 90.6574 172.076 25.695    75.0    0.15
##   ChlAugL BGAPCRFU BGAPCgL TurbFNU fDOMRFU fDOMQSU ODOsat ODOmgL   pH
## 1    0.32    -0.03    0.02    1.08    3.07    9.33  117.5   9.58 8.58
## 2    0.33    -0.02    0.03    1.08    3.07    9.33  117.5   9.58 8.58
## 3    0.36    -0.02    0.03    1.08    3.07    9.31  117.5   9.58 8.58
## 4    0.39    -0.02    0.02    1.05    3.06    9.31  117.6   9.58 8.59
## 5    0.40    -0.02    0.03    1.06    3.05    9.28  117.6   9.59 8.59
## 6    0.43    -0.02    0.03    1.05    3.05    9.26  117.6   9.59 8.59
##   Pressps XCO2Dppm_h XCO2Dppm_t XCH4Dppm_h XCH4Dppm_t TmpC_hy TempC_t
## 1   6.810    86.1409   81.04965    181.368         NA  25.664 25.5800
## 2   6.808    86.3325   80.96250    180.970    247.660  25.654 25.5660
## 3   6.806    85.7336   83.55935    181.669    233.257  25.643 25.5590
## 4   6.804    85.9029   83.99015    181.711    239.815  25.632 25.5488
## 5   6.804    85.6841   84.67335    182.032    234.358  25.623 25.5422
## 6   6.802    85.9059   84.87290    182.308    231.880  25.611 25.5326
##   SPCScm_h SPCScm_t ChlARFU_h ChlARFU_t ChlAgL_h ChlAgL_t BGAPCRFU_h
```

```
## 1     75.0    75.45      0.16      0.135      0.45    0.375       -0.01
## 2     75.0    75.45      0.16      0.135      0.45    0.350       -0.01
## 3     75.1    75.28      0.15      0.140      0.42    0.385       -0.01
## 4     75.1    75.19      0.15      0.140      0.41    0.370       -0.01
## 5     75.1    75.10      0.15      0.140      0.42    0.385       -0.02
## 6     75.1    75.10      0.15      0.140      0.42    0.380       -0.02
##   BGAPCRFU_t BGAPCgL_h BGAPCgL_t TrbFNU_h TrbFNU_t fDOMRFU_h fDOMRFU_t
## 1     -0.010      0.04     0.040     1.06    1.030      3.02     2.980
## 2     -0.010      0.04     0.005     1.05    0.855      3.01     2.970
## 3     -0.024      0.04     0.026     1.04    0.941      3.00     2.968
## 4     -0.031      0.03     0.009     0.92    0.824      2.99     2.962
## 5     -0.041      0.03     0.009     0.91    0.853      2.98     2.960
## 6     -0.034      0.03     0.016     0.90    0.873      2.98     2.968
##   fDOMQSU_h fDOMQSU_t ODOst_h ODOst_t ODOmgL_h ODOmgL_t pH_hyd pH_tau
## 1      9.19     9.070   117.6   117.6     9.60  9.60000   8.60  8.600
## 2      9.15     9.050   117.6   117.6     9.60  9.66835   8.60  8.675
## 3      9.13     9.026   117.6   117.6     9.60  9.62734   8.60  8.630
## 4      9.10     9.012   117.6   117.6     9.61  9.63734   8.61  8.640
## 5      9.07     8.998   117.6   117.6     9.61  9.62367   8.61  8.625
## 6      9.05     9.002   117.6   117.6     9.61  9.61000   8.61  8.610
##       CH4uM   CH4Sat   CH4M_hy  CH4St_h   CH4uM_t  CH4St_t    CO2uM
## 1 0.2221906 7945.911 0.2361346 8444.573        NA       NA 2.922544
## 2 0.2226015 7958.482 0.2356793 8426.042 0.3225304 11531.16 2.904789
## 3 0.2227614 7963.138 0.2366211 8458.588 0.3038137 10860.55 2.901845
## 4 0.2235733 7990.609 0.2367218 8460.543 0.3124161 11165.89 2.880868
## 5 0.2241721 8011.002 0.2371699 8475.489 0.3053455 10911.81 2.890373
## 6 0.2242392 8011.933 0.2375729 8488.340 0.3021722 10796.43 2.869197
##     CO2Sat  CO2M_hy  CO2St_h  CO2uM_t  CO2St_t
## 1 20.55636 2.722794 19.15137 2.561866 18.01945
## 2 20.42381 2.729874 19.19397 2.560072 18.00007
## 3 20.39928 2.711445 19.06081 2.642681 18.57742
## 4 20.24628 2.717542 19.09845 2.657032 18.67320
## 5 20.30949 2.711100 19.04981 2.679119 18.82509
## 6 20.15550 2.718818 19.09912 2.686124 18.86946
```

```r
models<-c("Nug", "Lin", "Sph") #Which models to try?

table<-ExtractRangeSill(LSG, colname='CH4uM_t', models=models, plot=T)
```




```r
print(table)
```

```
##   model     sill    range
## 1   Sph 0.243827 2081.026
```

Now let's apply it to multiple columns (variables) in one dataset

```r
columns<-c('TempC_t', 'SPCScm_t', 'TrbFNU_t', 'ODOmgL_t', 'pH_tau', 'fDOMQSU_t', 'CH4uM_t', 'CO2uM_t',

SemiVarList<-lapply(columns, function (x) ExtractRangeSill(SPdf=LSG, colname=x, models=models, plot=F))
```

```
## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : No convergence after 200 iterations: try different initial
## values?

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : No convergence after 200 iterations: try different initial
## values?

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : No convergence after 200 iterations: try different initial
## values?

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
```

```
## fit.ranges, : singular model in variogram fit

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : No convergence after 200 iterations: try different initial
## values?

## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges
## = fit.ranges, : No convergence after 200 iterations: try different initial
## values?

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : No convergence after 200 iterations: try different initial
## values?

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit

## Warning in fit.variogram(object, x, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : No convergence after 200 iterations: try different initial
## values?
```

```r
#Clean up output table
SemiVarTable<-ldply(SemiVarList, data.frame)
SemiVarTable$variable <- columns
SemiVarTable<-SemiVarTable[c('variable', 'model', 'sill', 'range')]
print(SemiVarTable)
```

```
##    variable model        sill     range
## 1   TempC_t   Lin  0.12240851  270.1061
## 2  SPCScm_t   Lin  3.92059807 2010.2453
## 3  TrbFNU_t   Lin  4.97359471 2244.8226
## 4  ODOmgL_t   Sph  0.15410605 2182.3036
## 5    pH_tau   Lin  0.14298146 2295.9883
## 6 fDOMQSU_t   Lin 36.19913486 1778.8389
## 7   CH4uM_t   Sph  0.24382697 2081.0256
## 8   CO2uM_t   Lin  2.74468262 2127.3073
## 9 ChlARFU_t   Lin  0.06144535 2214.1438
```

## Why is this useful?

Looking across lakes in northern Wisconsin, we see that variables differ in their spatial structure within lakes.

- Temperature tends to have the greatest semivariance ranges. Dissolved oxygen, chlorophyll, blue green algae, and turbidity have shorter ranges.
- Thus spatial structure occurs at finer scales for biotic varibles than physics!
- The same data as above, but ranges divided by lake size. A range ratio of 1 means the semivariance occured at the lake scale. Smaller ratios means the scale of variation occured at finer spatial scales
- In more than 50% of the lakes, the dominant scale of variation for temperature occurred at the lake scale (ratio=1).

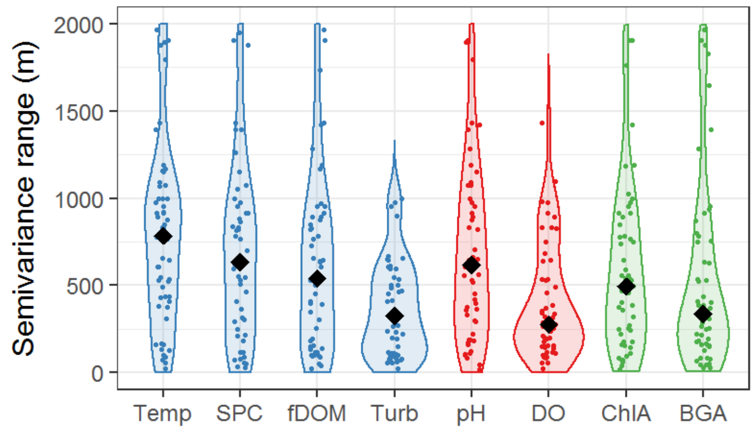



Figure 3: Semivariance ranges across lakes and variable types

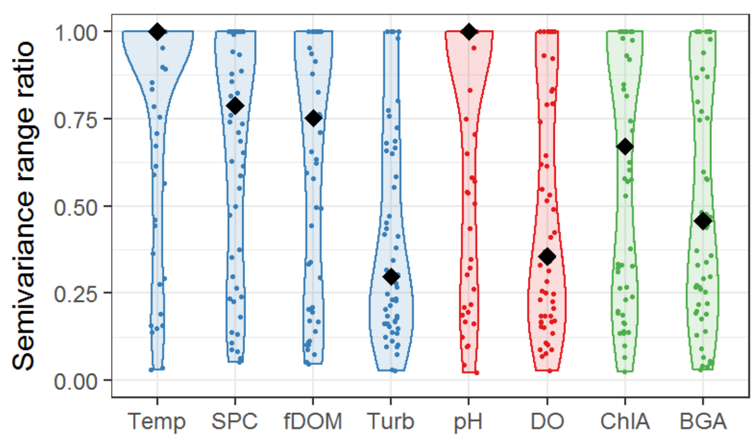
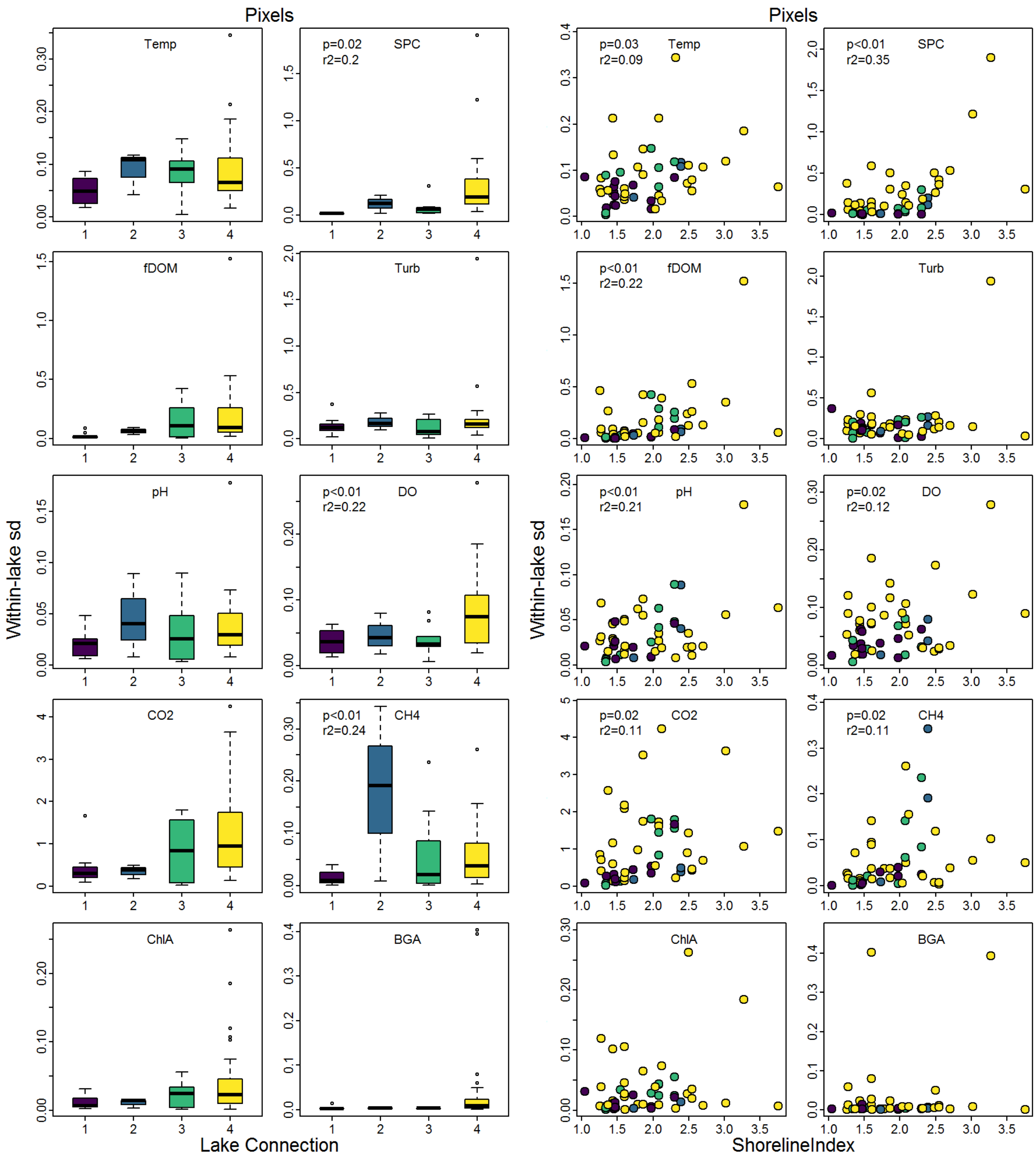



Figure 4: Semivariance range ratios across lakes and variable types

## Drivers of within lake variation among lakes?




- Best GLM models often select landscape position (or variables related to landscape position) and shoreline development index as lake scale predictors of within lake variability.
- Lakes lower in the landscape and lakes with more comlicated shorelines have greater within lake variation

## Take home points

- R is becoming more popular for spatial analysis, but it can be frustrating.
- Most modelling and interpolation methods require more inputs than may seem unnecessary. However, interpolation models probably should not be fit without considering the problem. I see in arcmap models you can click your way to success with defaults, but in reality you should consider all the assumptions with those choices.

El Fin