Interpolation Technique

Load libraries

```
library(dplyr)      # Data munging
library(fields)     # IDW weighting
library(reshape2)   # Final plot
library(ggplot2)    # Final plot
```

Build Sample Data

```
# Local stations
s1 <- data.frame(id = 1,
                 date = seq(as.Date("2000-01-15"), as.Date("2000-04-15"), "days"),
                 long = 10 + rep(rnorm(1, mean = 3, sd = 6)),
                 lat = 40 + rep(rnorm(1, mean = 3, sd = 6)),
                 temp = rnorm(92, mean = 67.5, sd = 15))

s2 <- data.frame(id = 2,
                 date = seq(as.Date("2000-01-15"), as.Date("2000-04-15"), "days"),
                 long = 10 + rep(rnorm(1, mean = 3, sd = 6)),
                 lat = 40 + rnorm(1, mean = 3, sd = 6),
                 temp = rnorm(92, mean = 67.5, sd = 15))

s3 <- data.frame(id = 3,
                 date = seq(as.Date("2000-01-15"), as.Date("2000-04-15"), "days"),
                 long = 10 + rep(rnorm(1, mean = 3, sd = 6)),
                 lat = 40 + rep(rnorm(1, mean = 3, sd = 6)),
                 temp = rnorm(92, mean = 67.5, sd = 15))

s4 <- data.frame(id = 4,
                 date = seq(as.Date("2000-01-15"), as.Date("2000-04-15"), "days"),
                 long = 10 + rep(rnorm(1, mean = 3, sd = 6)),
                 lat = 40 + rep(rnorm(1, mean = 3, sd = 6)),
                 temp = rnorm(92, mean = 67.5, sd = 15))

s5 <- data.frame(id = 5,
                 date = seq(as.Date("2000-01-15"), as.Date("2000-04-15"), "days"),
                 long = 10 + rep(rnorm(1, mean = 3, sd = 6)),
                 lat = 40 + rep(rnorm(1, mean = 3, sd = 6)),
                 temp = rnorm(92, mean = 67.5, sd = 15))

# Main Station monthly data temps
monthly_df <- data.frame(date = c("2000-01", "2000-02", "2000-03", "2000-04"),
                  long = 10 + rep(rnorm(1, mean = 3, sd = 6)),
                  lat = 40 + rep(rnorm(1, mean = 3, sd = 6)),
                  temp = c(70, 64, 72, 65))

# Build fine scale data frame
base <- data.frame(date = seq(as.Date("2000-01-15"), as.Date("2000-04-15"), "days"))
```

Spline interpolation

First, place the main station monthly data at the mid point for each month. Then run a spline through each of the mid points to interpolate daily data.

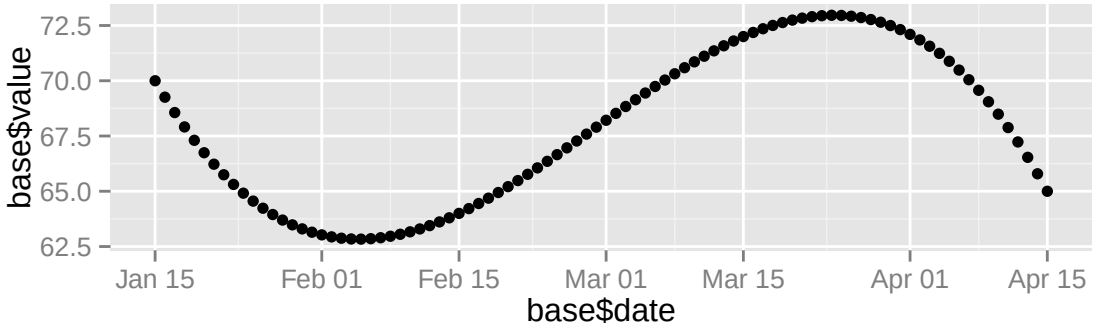
```
# Apply monthly data to mid points and build separate columns needed for spline
monthly_df$date <- as.Date(paste0(monthly_df$date, "-15"))
base$year <- as.numeric(substr(base$date, 1, 4))
base$month <- as.numeric(substr(base$date, 6, 7))
base$day <- as.numeric(substr(base$date, 9, 10))

# Merge monthly data to base mid points
base <- left_join(base, monthly_df, by = "date")

# Run a spline through each of the midpoints
values_by_date <- splinefun(unique(base[base$day == 15, 1]), base[!is.na(base$temp), 7 ])
base$value <- values_by_date(seq.Date(base$date[1], tail(base$date,1), by = "day"))

# Save spline data for plots below
spline_points <- base$value

# Plot spline interpolation
qplot(base$date, base$value)
```



Functions for relative anomaly interpolation

Two functions are built to interpolate the base data: (1) weights() returns the weights for the main station and 5 closest stations; (2) ra() returns the smoothed relative anomaly.

```
# Function to get weights
weights <- function(m_coord, s_coord){

  # Get distance from main station and local station
  d_mat <- rdist(m_coord, s_coord)

  # Apply IDW
  w_mat <- 1 / d_mat ^ 2

  # Check for inifinite and 0
  is_inf <- is.infinite(w_mat)
  has_infinite <- rowSums(is_inf) > 0
  w_mat[has_infinite, ] <- as.numeric(is_inf[has_infinite, ])

  # Keep 5 closest stations (you can change n to keep n closest stations)
  keep_n <- function(x, n = 5) ifelse(rank(x) > length(x)-n, x, 0)

  # Get weights for each local station
  w_mat <- t(apply(w_mat, 1, keep_n))
  w_mat <-  w_mat / rowSums(w_mat)
  return(w_mat)
}

# Function to return relative anomaly to apply to base
ra <- function(row_loc, weights){

  # Get 5 local station relative anomaly for each day
  s_values <- stack[row_loc:(row_loc+4),6]

  # Check for NA
  z <- is.na(s_values)

  # Return smoothed relative anomaly from local stations
  v <- (weights %*% ifelse(z, 0, s_values)) / (weights %*% !z)
  return(v)
}
```

Relative anomaly interpolation

And finally, stack the local stations, use weights() to get IDW, get relative anomaly from base data, stack again to get in daily order, and use sapply() to iterate through each day and apply smoothed relative anomaly.

```
# Stack local stations
stack <- rbind(s1, s2, s3, s4, s5)

# Get IDW weights
m_coord <- base[1,5:6]
s_coord <- data.frame(lat = unique(stack$lat), long = unique(stack$long))
```

```
weight <- weights(m_coord, s_coord)

# Find relative anomaly
s1$r <- s1$temp / base$value
s2$r <- s2$temp / base$value
s3$r <- s3$temp / base$value
s4$r <- s4$temp / base$value
s5$r <- s5$temp / base$value

# Stack again to get $r local stations
stack <- rbind(s1, s2, s3, s4, s5)

# Need to arrange by date so stack is in daily order
stack <- arrange(stack, date)

# Use ra() to smooth relative anomaly for each day and apply to base
newval <- sapply(1:nrow(base), function (i) base[i,8] * ra(row_loc = (5*(i-1)+1), weights = weight))
base$value <- newval

# Clean up base
base <- select(base, date, long, lat, value)
base[,2:3 ] <- lapply(base[,2:3], function(x) x[!is.na(x)][1])  # Fill in lat, long
```

Plot interpolated data and spline

```
# Plot with ggplot2
plot_data <- data.frame(date = base$date, spline = spline_points, value = base$value)

# Get in ggplot form
df <- melt(plot_data, id = "date")
qplot(date, value, data = df, col = variable)
```