child2

# Quantifying cold and long winters

The amount of energy required for heating is the function of the outside temperature.

Therefore to predict the energy needs left in a given (heating) season, we should make a prediction about the expected amount of energy required until the end of the season.

Thankfully things seem to be straightforward if we assume that the inside temperature of the building should be constant (it’s not but bear with me). Assume 20°C (68°F). Heat loss is linear to the difference in temperature between the inside and outside temperatures [link](http://hyperphysics.phy-astr.gsu.edu/hbase/thermo/heatloss.html).

Therefore when the outside temperature is 16°C we would need 4x the power to heat the building than if it would be 19°C. Also, this amount of energy needed for these two days is the same that we would need for heating for 3 days if it were degrees outside. In other words for a given heating season, we would add up the “missing degrees” until 20 for every day the outside temperature was less than 20°.

source( here::here( "inst", "function", "load\_stuff.r"))

Warning: package 'dplyr' was built under R version 4.3.2

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':  
  
 filter, lag

The following objects are masked from 'package:base':  
  
 intersect, setdiff, setequal, union

Attaching package: 'lubridate'

The following objects are masked from 'package:base':  
  
 date, intersect, setdiff, union

Warning: package 'nlme' was built under R version 4.3.2

Attaching package: 'nlme'

The following object is masked from 'package:dplyr':  
  
 collapse

act\_year <- 2023  
  
get\_approx\_meter <- approxfun(obs\_readings$Date,  
 obs\_readings$Value\_trf,   
 rule = 2, na.rm = TRUE)

Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):  
collapsing to unique 'x' values

get\_approx\_rate <- function(x) {  
 # dx get\_approx\_meter  
 h <- 1e-6  
 return((get\_approx\_meter(x + h) - get\_approx\_meter(x)) / h)  
}  
  
get\_avg\_temp <- function( df = obs\_days, var = "tavg", var\_time = "Date",  
 xmin, xmax) {  
 f <- approxfun( df[var\_time][[1]], df[var][[1]])  
 integrated <- integrate( f, subdivisions = 10000,  
 #stop.on.error = FALSE,  
 rel.tol = 0.1,  
 lower = xmin, upper = xmax)$value  
 duration <- xmax - xmin  
 return( integrated / duration)  
}  
  
  
obs\_days <- obs\_days %>%  
 mutate(Meter = get\_approx\_meter(Date),  
 Rate = get\_approx\_rate(Date)\*3600\*24) %>%  
 group\_by(ywint) %>%  
 mutate( Spent = Meter - min(Meter, na.rm = TRUE),  
 Spent\_perc = ifelse( year(Date) == act\_year,  
 Spent / 1730,  
 Spent / max(Spent, na.rm = TRUE)))

We have a pesky issue with missing data (sometimes every 2nd or 3rd date has results in the 90’s). The below step is quite rudimentary, doing linear interpolation between existing datapoints. This would capture the high autocorrelation between dates but it ain’t perfect.

# Step 1: Create a complete date sequence  
complete\_dates <-   
 data.frame(Date = seq(from = min(obs\_days$Date), to = max(obs\_days$Date), by = "day"))  
  
# Step 2: Expand your dataframe  
obs\_days\_complete <-   
 left\_join(complete\_dates, obs\_days,by="Date")  
   
  
# Assuming 'obs\_days\_complete' is your dataframe  
for (i in 2:ncol(obs\_days\_complete)) { # Starting from 2 to skip the Date column  
   
 # Extract dates and values for the current column  
 dat <- obs\_days\_complete$Date  
   
 if( colnames(obs\_days\_complete)[i] == "ywint") {   
 val <- as.numeric(obs\_days\_complete[, i])  
 } else {  
 val <- obs\_days\_complete[, i]  
 }  
  
 # Create the interpolation function only for non-NA values  
 valid\_idx <- !is.na(val)  
 if(sum(valid\_idx) > 1) { # Ensure there are at least two points for interpolation  
 fun. <- approxfun(x = dat[valid\_idx], y = val[valid\_idx], rule = 2)  
   
 # Apply the function to interpolate NA values  
 obs\_days\_complete[, i] <- ifelse(is.na(val), fun.(dat), val)  
 }  
}  
  
obs\_days\_complete <- obs\_days\_complete %>%  
 arrange(Date) %>%  
 mutate(tavg\_capped = ifelse(tavg < 20, tavg, 20),  
 tavg\_low\_cumul = 0)  
  
  
pb <- txtProgressBar(style=3)

|   
 | | 0%

for (i in 2:nrow(obs\_days\_complete)) {  
 if ( obs\_days\_complete$day\_in\_wint[i] >= obs\_days\_complete$day\_in\_wint[i-1]) {  
 obs\_days\_complete$tavg\_low\_cumul[i] <-   
 obs\_days\_complete$tavg\_low\_cumul[i - 1] - obs\_days\_complete$tavg\_capped[i] + 20  
 } else {  
 # in case of a new year, its automatically zero  
 obs\_days\_complete$tavg\_low\_cumul[i] <- obs\_days\_complete$tavg\_capped[i] - 20  
 }  
 setTxtProgressBar(pb,i/nrow(obs\_days\_complete))  
}

|   
 | | 1%  
 |   
 |= | 1%  
 |   
 |= | 2%  
 |   
 |== | 2%  
 |   
 |== | 3%  
 |   
 |== | 4%  
 |   
 |=== | 4%  
 |   
 |=== | 5%  
 |   
 |==== | 5%  
 |   
 |==== | 6%  
 |   
 |===== | 6%  
 |   
 |===== | 7%  
 |   
 |===== | 8%  
 |   
 |====== | 8%  
 |   
 |====== | 9%  
 |   
 |======= | 9%  
 |   
 |======= | 10%  
 |   
 |======= | 11%  
 |   
 |======== | 11%  
 |   
 |======== | 12%  
 |   
 |========= | 12%  
 |   
 |========= | 13%  
 |   
 |========= | 14%  
 |   
 |========== | 14%  
 |   
 |========== | 15%  
 |   
 |=========== | 15%  
 |   
 |=========== | 16%  
 |   
 |============ | 16%  
 |   
 |============ | 17%  
 |   
 |============ | 18%  
 |   
 |============= | 18%  
 |   
 |============= | 19%  
 |   
 |============== | 19%  
 |   
 |============== | 20%  
 |   
 |============== | 21%  
 |   
 |=============== | 21%  
 |   
 |=============== | 22%  
 |   
 |================ | 22%  
 |   
 |================ | 23%  
 |   
 |================ | 24%  
 |   
 |================= | 24%  
 |   
 |================= | 25%  
 |   
 |================== | 25%  
 |   
 |================== | 26%  
 |   
 |=================== | 26%  
 |   
 |=================== | 27%  
 |   
 |=================== | 28%  
 |   
 |==================== | 28%  
 |   
 |==================== | 29%  
 |   
 |===================== | 29%  
 |   
 |===================== | 30%  
 |   
 |===================== | 31%  
 |   
 |====================== | 31%  
 |   
 |====================== | 32%  
 |   
 |======================= | 32%  
 |   
 |======================= | 33%  
 |   
 |======================= | 34%  
 |   
 |======================== | 34%  
 |   
 |======================== | 35%  
 |   
 |========================= | 35%  
 |   
 |========================= | 36%  
 |   
 |========================== | 36%  
 |   
 |========================== | 37%  
 |   
 |========================== | 38%  
 |   
 |=========================== | 38%  
 |   
 |=========================== | 39%  
 |   
 |============================ | 39%  
 |   
 |============================ | 40%  
 |   
 |============================ | 41%  
 |   
 |============================= | 41%  
 |   
 |============================= | 42%  
 |   
 |============================== | 42%  
 |   
 |============================== | 43%  
 |   
 |============================== | 44%  
 |   
 |=============================== | 44%  
 |   
 |=============================== | 45%  
 |   
 |================================ | 45%  
 |   
 |================================ | 46%  
 |   
 |================================= | 46%  
 |   
 |================================= | 47%  
 |   
 |================================= | 48%  
 |   
 |================================== | 48%  
 |   
 |================================== | 49%  
 |   
 |=================================== | 49%  
 |   
 |=================================== | 50%  
 |   
 |=================================== | 51%  
 |   
 |==================================== | 51%  
 |   
 |==================================== | 52%  
 |   
 |===================================== | 52%  
 |   
 |===================================== | 53%  
 |   
 |===================================== | 54%  
 |   
 |====================================== | 54%  
 |   
 |====================================== | 55%  
 |   
 |======================================= | 55%  
 |   
 |======================================= | 56%  
 |   
 |======================================== | 56%  
 |   
 |======================================== | 57%  
 |   
 |======================================== | 58%  
 |   
 |========================================= | 58%  
 |   
 |========================================= | 59%  
 |   
 |========================================== | 59%  
 |   
 |========================================== | 60%  
 |   
 |========================================== | 61%  
 |   
 |=========================================== | 61%  
 |   
 |=========================================== | 62%  
 |   
 |============================================ | 62%  
 |   
 |============================================ | 63%  
 |   
 |============================================ | 64%  
 |   
 |============================================= | 64%  
 |   
 |============================================= | 65%  
 |   
 |============================================== | 65%  
 |   
 |============================================== | 66%  
 |   
 |=============================================== | 66%  
 |   
 |=============================================== | 67%  
 |   
 |=============================================== | 68%  
 |   
 |================================================ | 68%  
 |   
 |================================================ | 69%  
 |   
 |================================================= | 69%  
 |   
 |================================================= | 70%  
 |   
 |================================================= | 71%  
 |   
 |================================================== | 71%  
 |   
 |================================================== | 72%  
 |   
 |=================================================== | 72%  
 |   
 |=================================================== | 73%  
 |   
 |=================================================== | 74%  
 |   
 |==================================================== | 74%  
 |   
 |==================================================== | 75%  
 |   
 |===================================================== | 75%  
 |   
 |===================================================== | 76%  
 |   
 |====================================================== | 76%  
 |   
 |====================================================== | 77%  
 |   
 |====================================================== | 78%  
 |   
 |======================================================= | 78%  
 |   
 |======================================================= | 79%  
 |   
 |======================================================== | 79%  
 |   
 |======================================================== | 80%  
 |   
 |======================================================== | 81%  
 |   
 |========================================================= | 81%  
 |   
 |========================================================= | 82%  
 |   
 |========================================================== | 82%  
 |   
 |========================================================== | 83%  
 |   
 |========================================================== | 84%  
 |   
 |=========================================================== | 84%  
 |   
 |=========================================================== | 85%  
 |   
 |============================================================ | 85%  
 |   
 |============================================================ | 86%  
 |   
 |============================================================= | 86%  
 |   
 |============================================================= | 87%  
 |   
 |============================================================= | 88%  
 |   
 |============================================================== | 88%  
 |   
 |============================================================== | 89%  
 |   
 |=============================================================== | 89%  
 |   
 |=============================================================== | 90%  
 |   
 |=============================================================== | 91%  
 |   
 |================================================================ | 91%  
 |   
 |================================================================ | 92%  
 |   
 |================================================================= | 92%  
 |   
 |================================================================= | 93%  
 |   
 |================================================================= | 94%  
 |   
 |================================================================== | 94%  
 |   
 |================================================================== | 95%  
 |   
 |=================================================================== | 95%  
 |   
 |=================================================================== | 96%  
 |   
 |==================================================================== | 96%  
 |   
 |==================================================================== | 97%  
 |   
 |==================================================================== | 98%  
 |   
 |===================================================================== | 98%  
 |   
 |===================================================================== | 99%  
 |   
 |======================================================================| 99%  
 |   
 |======================================================================| 100%

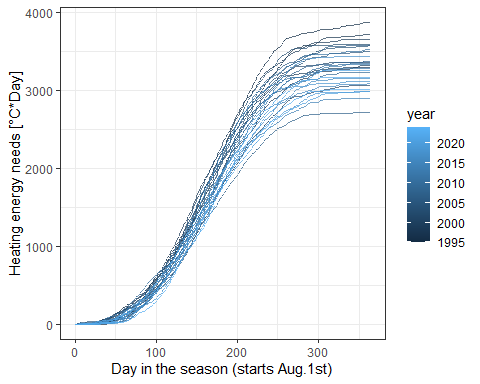
close(pb)

obs\_days\_complete <- obs\_days\_complete %>%  
 mutate(ywint = as.factor( ywint))  
  
# There are some edge cases I have to take care of (due to leap years)  
obs\_days\_complete$tavg\_low\_cumul[obs\_days\_complete$ywint==1] <- NA  
obs\_days\_complete$tavg\_low\_cumul[obs\_days\_complete$day\_in\_wint > 180 &  
 obs\_days\_complete$tavg\_low\_cumul < 100] <- NA

obs\_days\_complete %>%  
 ggplot(aes(x = day\_in\_wint, y = tavg\_low\_cumul  
 )) +  
 theme\_bw() +  
 geom\_line(alpha=.7, mapping = aes(  
 color = year  
 ,fill=year  
 ,group = factor(ywint))) +  
 labs( x = "Day in the season (starts Aug.1st)",  
 y = "Heating energy needs [°C\*Day]")

Warning in geom\_line(alpha = 0.7, mapping = aes(color = year, fill = year, :  
Ignoring unknown aesthetics: fill

Warning: Removed 214 rows containing missing values (`geom\_line()`).



I personally like this approach very much, but of course we are neglecting:

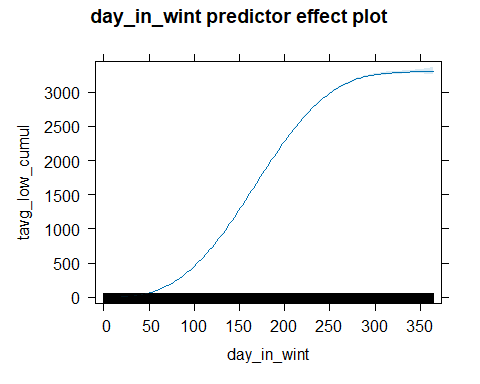
* The effects of solar radiation, eg. sunlight heats up the building a bit,
* the building’s heat capacity, eg. after a hot summer’s day, it takes about 3 days for the walls to cool

For simplicity and to retain my sanity, lets also assume that at each timepoint (day), the distribution of ‘energy needs’ is normal.

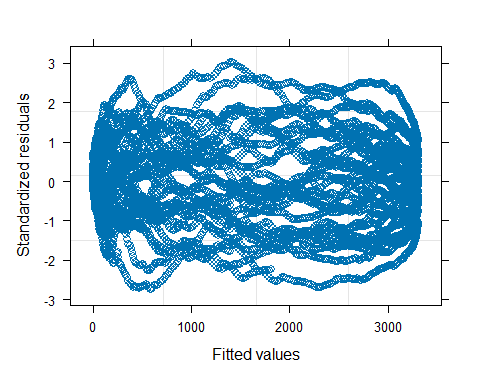
|  |
| --- |
| Important |
| The evolution of the energy needs *may* be thought of as logistic growth but I’m not exploring that at this time. |

I’d like to fit a *linear model* for the energy needs with a natural spline term containing the elapsed days. To capture the apparent heteroscedasticity, I’d opt for a *generalized least squares* model with an *exponential variance structure* (meaning that the residuals are understood to have a larger variance when more days have elapsed). Its not perfect, since at the end of the season (june, july) not much heating is going on and therefore variance tends to taper off there.

mod\_cum <- gls(tavg\_low\_cumul ~ ns(day\_in\_wint,df=8),  
 obs\_days\_complete,  
 weights=varExp(form = ~ day\_in\_wint),  
 na.action = na.omit)  
  
#summary(mod\_cum)  
mod\_cum %>% effects::predictorEffects(partial.residuals=FALSE) %>% plot



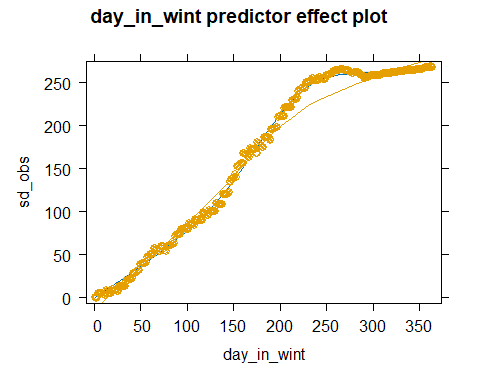
plot(mod\_cum)



I could not, for the life of me got the *prediction intervals* correctly out of the model which would expand over the course of the elapsed days. One potential solution would be the package *AICcmodavg* but it threw an error (possibly because I used *tibbles* for my data; solution is trivial, but not going to implement it).

As a “petty craftsman” kind of approach, I calculated standard deviations for all days and then fitted a *linear model* over it with a natural spline term capturing the evolution of standard deviation in heating needs over the course of a season.

sds <- data.frame(  
 day\_in\_wint = 1:364,  
 sd\_obs = NA  
)  
  
for (i in 1:nrow(sds)) {  
 sds$sd\_obs[i] <-   
 obs\_days\_complete %>%   
 filter(day\_in\_wint == sds$day\_in\_wint[i]) %>%   
 .$tavg\_low\_cumul %>%   
 sd(na.rm = TRUE)  
}  
mod\_cum\_obs <- lm( sd\_obs ~ ns(day\_in\_wint,df=5) - 1  
 ,sds)  
  
sds$sd\_pred <- predict(mod\_cum\_obs)  
sds$mean\_pred <- predict(mod\_cum, newdata = sds)  
  
mod\_cum\_obs %>%   
 effects::predictorEffects(  
 partial.residuals = TRUE  
 ) %>%   
 plot()



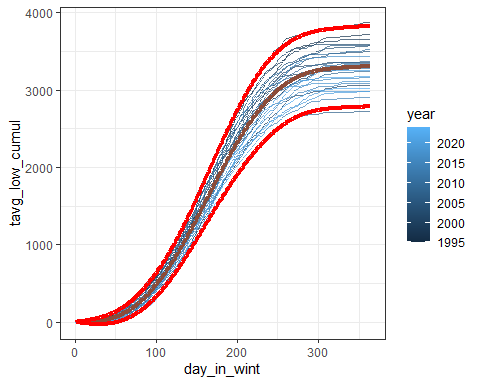
Now we have a mean and a variance for the cumulative heating energy needs on any given day (at my location)! I can then assign a z-score for each result through the years.

sd\_fun <- approxfun(sds$day\_in\_wint,sds$sd\_pred)  
mean\_fun <- approxfun(sds$day\_in\_wint,sds$mean\_pred)  
  
return\_std\_tmp <- function(day\_act, tmp\_act) {  
 m <- mean\_fun(day\_act)  
 s <- sd\_fun(day\_act)  
 return(list(  
 z = (tmp\_act - m) / s,  
 z\_perc = pnorm((tmp\_act - m) / s)  
 ))  
}  
  
obs\_days\_complete <-   
 obs\_days\_complete %>%  
 mutate(z = return\_std\_tmp(day\_in\_wint,tavg\_low\_cumul)[["z"]],  
 z\_perc = return\_std\_tmp(day\_in\_wint,tavg\_low\_cumul)[["z\_perc"]])

obs\_days\_complete %>%  
 ggplot(aes(x = day\_in\_wint, y = tavg\_low\_cumul  
 )) +  
 theme\_bw() +  
 geom\_line(alpha=.7, mapping = aes(  
 color = year  
 ,fill=year  
 ,group = factor(ywint))) +  
 geom\_line(data = sds, mapping = aes(y = mean\_pred),color='salmon4',linewidth=1.5) +  
 geom\_line(data = sds, mapping = aes(y = mean\_pred + 1.96 \* sd\_pred),color='red',linewidth=1.5) +  
 geom\_line(data = sds, mapping = aes(y = mean\_pred - 1.96 \* sd\_pred),color='red',linewidth=1.5)

Warning in geom\_line(alpha = 0.7, mapping = aes(color = year, fill = year, :  
Ignoring unknown aesthetics: fill

Warning: Removed 214 rows containing missing values (`geom\_line()`).



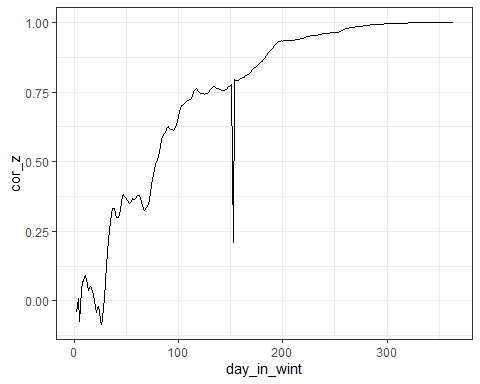
# Estimating total energy needs left on a given day

This is all well and good, and this translates to an estimated total energy need at the beginning of a season, but we are mainly interested in the *conditional* energy need for the total season if we have observed the cumulative energy need up to day “*d”.*

I start with investigating how do energy need values observed on a given day *correlate* to the *total* energy need for a given season. Specifically, what was the correlation between the z-scores of a given day vs. the last day in a season.

final\_z <-   
 obs\_days\_complete %>%   
 filter(day\_in\_wint == 364) %>%   
 mutate(id = 1:n())  
sds$cor\_z <- NA  
for (i in 1:364) {  
 act\_z <-   
 obs\_days\_complete %>%   
 filter(day\_in\_wint == i)   
   
 if ( i < 155) {  
 act\_z <- act\_z %>%  
 mutate(id=2:(n()+1)) %>%  
 left\_join(y=final\_z,by="id")  
 } else {  
 act\_z <- act\_z %>%  
 mutate(id=1:n()) %>%  
 left\_join(y=final\_z,by="id")  
 }  
   
 sds$cor\_z[i] <- cor(act\_z$z.x,act\_z$z.y,use="pairwise.complete.obs")  
}  
  
sds %>%  
 ggplot(aes(x=day\_in\_wint,y=cor\_z)) +  
 theme\_bw() +  
 geom\_line() #+

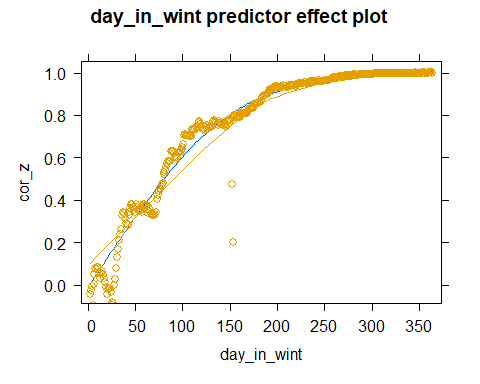
Warning: Removed 1 row containing missing values (`geom\_line()`).



#geom\_smooth(alpha=.2)

The spike around day 150 is due to leap years (probably); didn’t bother to fix it, as it didn’t bother the results much. I fit a linear model with a spline for this relationship, like before.

mod\_cor <- lm(cor\_z~ns(day\_in\_wint,df=3) - 1,sds)  
#summary(mod\_cor)  
sds$cor\_z\_pred <- predict(mod\_cor, newdata = sds)  
sds <- sds %>% mutate(cor\_z\_pred = cor\_z\_pred/max(cor\_z\_pred,na.rm=TRUE))  
  
mod\_cor %>% effects::predictorEffects(partial.residuals=TRUE) %>% plot()



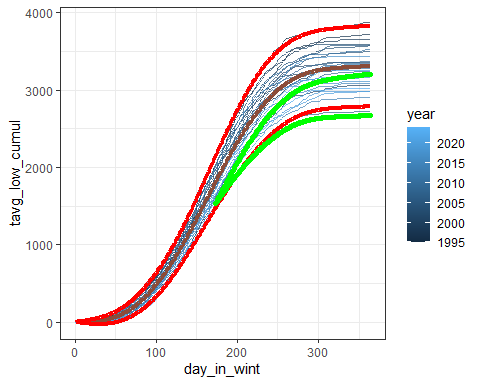
Sooo…. we are searching for the *Y* normal distribution, which is correlated by *c* with the normal distribution *X*, then

Lets fix day n the season as *173* and the cumulated energy need thus far as *1530* (beginning of January, 2024). I’m not a 100% that I’ve implemented the intermediate values the right way, but I am sure about the end state (at Day 365). Level of significance is at 5% (hence the 1.95x SD part).

conf\_lev <- 1.96  
day\_in\_wint\_act <- 173  
cum\_t\_act <- 1530  
  
cor\_fun <- approxfun(sds$day\_in\_wint,sds$cor\_z\_pred)  
cor\_act <- cor\_fun( day\_in\_wint\_act)  
sd\_expected <- sqrt( 1 - cor\_act^2)   
z\_act <- return\_std\_tmp(day\_in\_wint\_act, cum\_t\_act)$z  
z\_expected <- cor\_act \* z\_act  
  
# lower\_expected\_std <- z\_expected - conf\_lev \* sd\_expected  
# upper\_expected\_std <- z\_expected + conf\_lev \* sd\_expected  
  
expecteds <-   
 data.frame(day\_in\_wint = day\_in\_wint\_act:364) %>%  
 left\_join(y=sds,by="day\_in\_wint") %>%  
 mutate(prop\_var = (cor\_z\_pred - cor\_act) / (1 - cor\_act),  
 sd\_act = sd\_expected \* prop\_var,  
 z\_act. = seq(z\_act,z\_expected,length.out=n()),  
 lower\_expected = mean\_fun(day\_in\_wint) + (z\_act. - sd\_act\*conf\_lev)\*sd\_fun(day\_in\_wint),  
 upper\_expected = mean\_fun(day\_in\_wint) + (z\_act. + sd\_act\*conf\_lev)\*sd\_fun(day\_in\_wint)  
 )  
  
obs\_days\_complete %>%  
 ggplot(aes(x = day\_in\_wint, y = tavg\_low\_cumul  
 )) +  
 theme\_bw() +  
 geom\_line(alpha=.7, mapping = aes(  
 color = year  
 ,fill=year  
 ,group = factor(ywint))) +  
 geom\_line(data = sds, mapping = aes(y = mean\_pred),  
 color='salmon4',linewidth=1.5) +  
 geom\_line(data = sds, mapping = aes(y = mean\_pred + 1.96 \* sd\_pred),  
 color='red',linewidth=1.5) +  
 geom\_line(data = sds, mapping = aes(y = mean\_pred - 1.96 \* sd\_pred),  
 color='red',linewidth=1.5) +  
 geom\_point(data = expecteds,mapping=aes(y=lower\_expected),color = "green") +  
 geom\_point(data = expecteds,mapping=aes(y=upper\_expected),color = "green") +  
 geom\_point(mapping = aes( x = day\_in\_wint\_act, y =cum\_t\_act),color="green")

Warning in geom\_line(alpha = 0.7, mapping = aes(color = year, fill = year, :  
Ignoring unknown aesthetics: fill

Warning: Removed 214 rows containing missing values (`geom\_line()`).



I can therefore construct an interval estimate of how much energy I still need to burn in this heating season. I prefer to see them as percents out of a total energy budget.

# Percent heating req. elapsed (lower bound)  
expecteds %>%  
 last %>%  
 mutate(result = cum\_t\_act / upper\_expected) %>%  
 .$result

[1] 0.4769238

# Percent heating req. elapsed (upper bound)  
expecteds %>%  
 last %>%  
 mutate(result = cum\_t\_act / lower\_expected) %>%  
 .$result

[1] 0.5726802

It seems that up until this point we (should) have used up 47-57% of our total energy budget for this season.