Online Appendix: Code

The following code (in R) can be used to re-create any result from the model.

```
create_env <- function(M, K, U, E, test_rows=2) {
  main_effects <- paste0('x',1:M)
  interactions <- apply(combn(main_effects, 2), 2, paste, collapse='*')
  all_terms    <- c(1, main_effects, interactions)
  terms        <- all_terms[sort(sample(1:length(all_terms), K))]
  coeffs       <- rnorm(K)
  fmla         <- paste(apply(rbind(coeffs, terms), 2, paste, collapse='*'),
                        collapse=' + ')
  nrows        <- E + test_rows
  Xs           <- as.data.frame(matrix(rnorm(nrows*M), nrow=nrows, ncol=M,
                                       dimnames=list(1:nrows, main_effects)))
  Ys_no_noise  <- eval(parse(text=fmla), envir=as.list(Xs))
  Ys           <- Ys_no_noise * rnorm(nrows, mean=1, sd=U)
  list(all_terms=all_terms, terms=terms, coeffs=coeffs, fmla=fmla, U=U, Xs=Xs[1:E,],
       Ys=Ys[1:E], test_Xs=Xs[(E+1):(E+test_rows),], test_Ys=Ys[(E+1):(E+test_rows)])
}
# Example usage:
# env <- create_env(M=3, K=2, U=0.5, E=7, test_rows=2); env

OLS <- function(X, Y) tryCatch(qr.solve(X, Y), error=function(e)
{ message('Ill-conditioned matrix'); dput(cbind(X,Y)); rep(NA, ncol(X)) })

estimate_repr <- function(env, Kprime, I) {
  if (I==0) {
    terms <- env$all_terms[sort(sample(1:length(env$all_terms), Kprime))]
  } else {
    terms <- c(env$terms[order(-abs(env$coeffs))],
               sample(setdiff(env$all_terms, env$terms)))[1:Kprime]
  }
  Xs_obs  <- eval(parse(text=paste0('cbind(',paste(terms, collapse=','),')')),
                  envir=as.list(env$Xs))
  colnames(Xs_obs) <- terms
  if (length(terms)==1 && (terms=='1')) Xs_obs <- matrix(Xs_obs, length(env$Ys))
  coeffs  <- OLS(Xs_obs, env$Ys)
  fmla    <- paste(apply(rbind(coeffs, terms), 2, paste, collapse='*'), collapse=' + ')
  pred_Ys <- eval(parse(text=fmla), envir=as.list(env$test_Xs))
  if (length(terms)==1 && (terms=='1')) pred_Ys <- matrix(pred_Ys, nrow(env$test_Xs))
  list(terms=terms, Xs_obs=Xs_obs, coeffs=coeffs, fmla=fmla, pred_Ys=pred_Ys)
}
# Example usage:
# repr <- estimate_repr(env, Kprime=1, I=1); repr

sim <- function(M, K, U, Kprimes, E, I, nsims, ntests, rawdata=FALSE) {
  onesim <- function(x) {
    repeat { # redraw if matrix is ill-conditioned (1-in-10 million chance)
      env  <- create_env(M, K, U, E, test_rows=ntests)
      repr <- estimate_repr(env, Kp, I)
      if (is.numeric(repr$coeffs)) break()
    }
    data.frame(test_Y=env$test_Ys, pred_Y=repr$pred_Ys)
  }
  res <- list()
  for (Kp in Kprimes) {
    D <- do.call(rbind, lapply(1:nsims, onesim))
    D <- cbind(expand.grid(Kprime=Kp,ntest=1:ntests,nsim=1:nsims), D)
```

```
    D$decision     <- 1*(D$pred_Y>0)
    D$actual_perf  <- 1*(D$pred_Y>0) * D$test_Y
    D$optimal_perf <- 1*(D$test_Y>0) * D$test_Y
    perf <- (mean(D$actual_perf)-mean(D$test_Y))/(mean(D$optimal_perf)-mean(D$test_Y))
    res[[Kp]] <- c(perf=perf)
  }
  res <- do.call(rbind, res); rownames(res) <- Kprimes
  if (rawdata) { D } else { res }
}

# Replicate any line in Figure 2
K=4; U=0.5; E=7; I=1
D <- sim(M=3, K=K, U=U, Kprimes=1:7, E=E, I=I, nsims=1000, ntests=100)
plot(D, ylim=c(0,1), type='l', xlab="Representational Complexity (K')", ylab='Performance',
     cex.main=1, font.main=1, main=sprintf('K=%s   U=%s   E=%s   I=%s',K,U,E,I))
```