Component response rate variation drives stability in large complex systems

Supporting information

Brad Duthie

Code and simulations underlying Fig. 1

The sample M used for the eigenvalue distributions in Fig. 1 of the text is available on GitHub, and was produced with by running the following function.

```
find_bgamma <- function(S = 200, C = 0.05, Osd = 0.4, iters = 10000){
    while(iters > 0){
        A_dat  <- rnorm(n = S * S, mean = 0, sd = Osd);
        A_mat  <- matrix(data = A_dat, nrow = S);
        C_dat  <- rbinom(n = S * S, size = 1, prob = C);
        C_mat  <- matrix(data = C_dat, nrow = S, ncol = S);
        A_mat  <- A_mat * C_mat;
        gammas <- c(rep(1.95, S/2), rep(0.05, S/2))
        mu_gam <- mean(gammas);
        diag(A_mat) <- -1;
        A1     <- gammas * A_mat;
        A0     <- mu_gam * A_mat;
        A0_e   <- eigen(A0)$values;
        A0_r   <- Re(A0_e);
        A0_i   <- Im(A0_e);
        A1_e   <- eigen(A1)$values;
        A1_r   <- Re(A1_e);
        A1_i   <- Im(A1_e);
        if(max(A0_r) >= 0 & max(A1_r) < 0){
            return(list(A0 = A0, A1 = A1));
            break;
        }
        print(iters);
        iters <- iters - 1;
    }
}
```

The above function terminates when a matrix M is found that is not stable when all component response rates are set to $\gamma = 1$, but is stable when half of component response rates are 1.95 and half are 0.05. The function is used to illustrate the concept of how a fast versus slow component responses can cause a system to become stable. Simulations were run for `iter = 1000000`, but terminated once an acceptable `A0` and `A1` were found. The code below plots the eigenvalue distributions of `A0` and `A1` in panels **a** and **b**, respectively.

The plot itself can be recreated with the code below.

```
par(mfrow = c(1, 2), mar = c(0.5, 0.5, 0.5, 0.5), oma = c(5, 5, 0, 0));
plot(A0_r, A0_i, xlim = c(-3.7, 0.3), ylim = c(-2, 2), pch = 4, cex = 0.7,
     xlab = "", ylab = "", cex.lab = 1.5, cex.axis = 1.5, asp = 1);
vl <- seq(from = 0, to = 2*pi, by = 0.001);
A0x0 <- sqrt(200) * sd(A0vec) * cos(vl) + mean(diag(A0));
A0y0 <- sqrt(200) * sd(A0vec) * sin(vl);
```

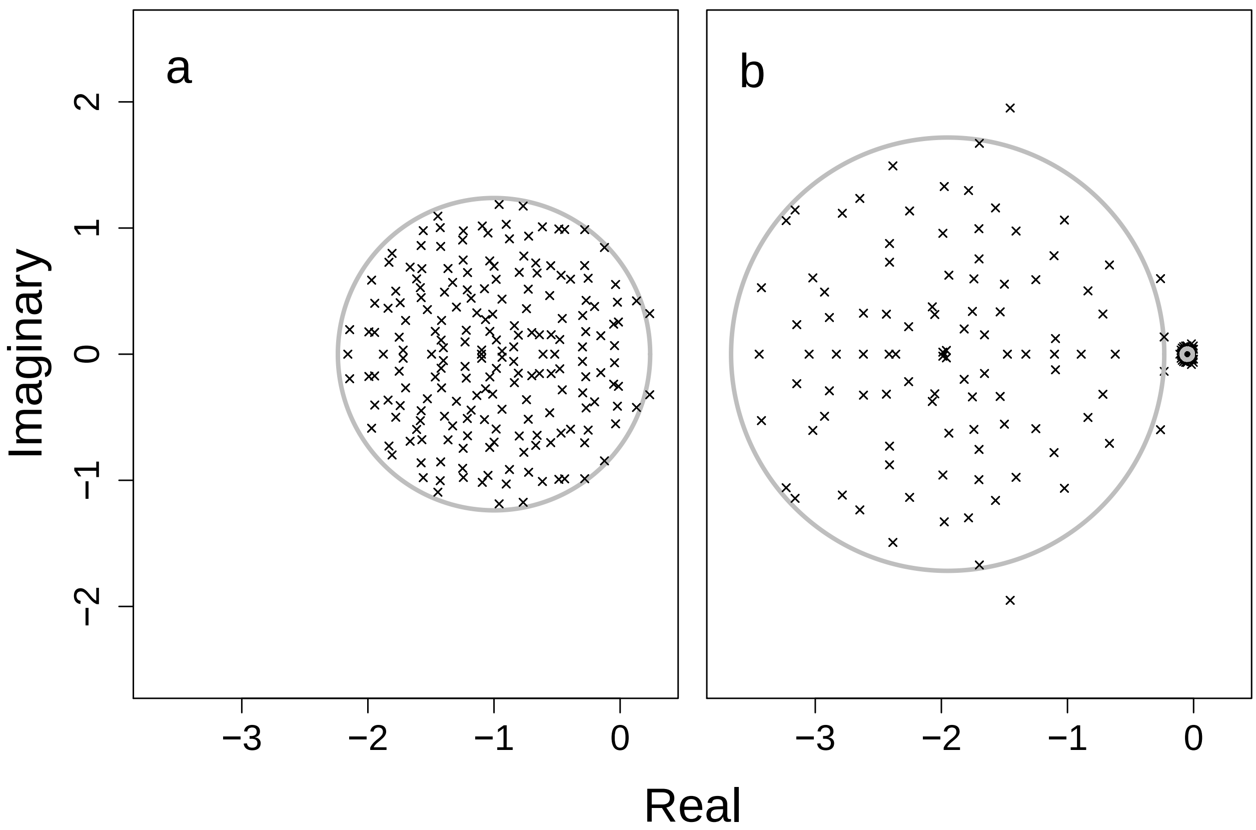
```
text(x = -3.5, y = 2.25, labels = "a", cex = 2);
points(x = A0x0, y = A0y0, type = "l", lwd = 3, col = "grey");
points(A0_r, A0_i, pch = 4, cex = 0.7);

plot(A1_r, A1_i, xlim = c(-3.7, 0.3), ylim = c(-2, 2), pch = 4, cex = 0.7,
     xlab = "", ylab = "", cex.lab = 1.5, cex.axis = 1.5, asp = 1,
     col = "black", yaxt = "n");

vl <- seq(from = 0, to = 2*pi, by = 0.001);
A0x1a <- sqrt(100) * sd(A1vec[fhalf]) * cos(vl) + mean(diag(A1)[1:100]);
A0y1a <- sqrt(100) * sd(A1vec[fhalf]) * sin(vl);
points(x = A0x1a, y = A0y1a, type = "l", lwd = 3, col = "grey");
A0x1b <- sqrt(100) * sd(A1vec[shalf]) * cos(vl) + mean(diag(A1)[101:200]);
A0y1b <- sqrt(100) * sd(A1vec[shalf]) * sin(vl);
points(x = A0x1b, y = A0y1b, type = "l", lwd = 3, col = "grey");

points(A1_r[1:100], A1_i[1:100],pch = 4, cex = 0.7);

text(x = -3.5, y = 2.25, labels = "b", cex = 2);
mtext(side = 1, "Real", outer = TRUE, line = 3, cex = 2);
mtext(side = 2, "Imaginary", outer = TRUE, line = 2.5, cex = 2);
```



To find out how frequently M was stable given that all $\gamma = 1$ versus $\gamma = \{1.95, 0.05\}$, the function below was created.

```
stab_bgamma <- function(S = 200, C = 0.05, Osd = 0.4, iters = 10000){
    ress     <- matrix(data = 0, nrow = iters, ncol = 2);
    A0_count <- 0;
```

```
    A1_count <- 0;
    while(iters > 0){
        A_dat  <- rnorm(n = S * S, mean = 0, sd = Osd);
        A_mat  <- matrix(data = A_dat, nrow = S);
        C_dat  <- rbinom(n = S * S, size = 1, prob = C);
        C_mat  <- matrix(data = C_dat, nrow = S, ncol = S);
        A_mat  <- A_mat * C_mat;
        gammas <- c(rep(1.95, S/2), rep(0.05, S/2))
        mu_gam <- mean(gammas);
        diag(A_mat) <- -1;
        A1     <- gammas * A_mat;
        A0     <- mu_gam * A_mat;
        A0_e   <- eigen(A0)$values;
        A0_r   <- Re(A0_e);
        A0_i   <- Im(A0_e);
        A1_e   <- eigen(A1)$values;
        A1_r   <- Re(A1_e);
        A1_i   <- Im(A1_e);
        if(max(A0_r) < 0){
            ress[iters, 1] <- 1;
            A0_count       <- A0_count + 1;
            }
        if(max(A1_r) < 0){
            ress[iters, 2] <- 1;
            A1_count       <- A1_count + 1;
        }
        print(c(iters, A0_count, A1_count));
        iters <- iters - 1;
    }
    return(ress);
}
```

The function above was run for `iters = 1000000`, and the resulting matrix `ress` was returned. Each row of `ress` represents a single M given $\gamma = 1$ (column 1) versus $\gamma = \{1.95, 0.05\}$ (column 2). Values of 0 indicate that M was found to be unstable (at least one real component of its eigenvalues greater than or equal to zero), whereas values of 1 indicate that M was found to be stable (all real components of eigenvalues are negative). The frequencies of stable M were 0 given $\gamma = 1$ and 0 given $\gamma = \{1.95, 0.05\}$, as reported in the main text and legend of Fig. 1.

Code and simulations underlying Fig. 2

Figure 2 of the main text shows how eigenvalue distributions in a system where $S = 1000$, $C = 1$, and $\sigma = 0.4$. Eigenvalues can be reproduced using the code below for when $\gamma = 1$ (panel a) and $\gamma \sim \mathcal{U}(0, 2)$ (panel b).

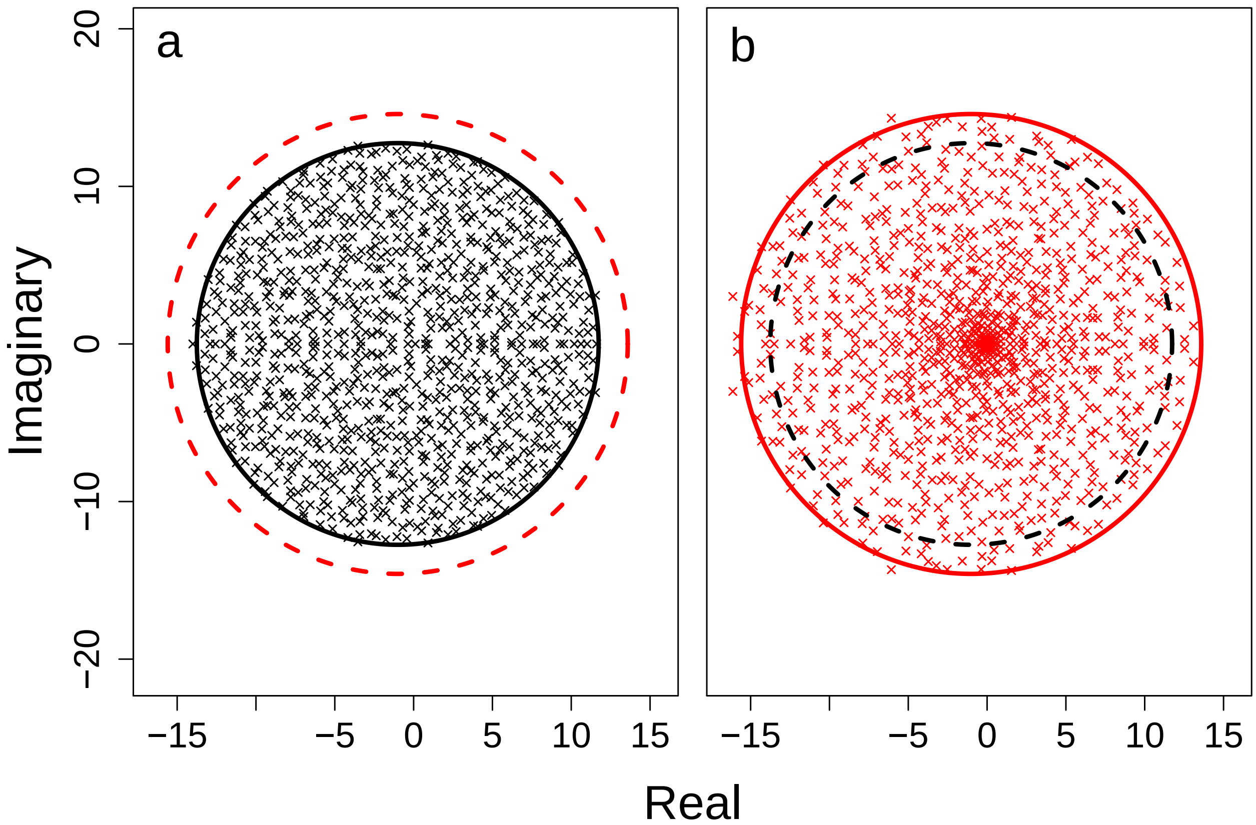
```
A_comp <- NULL;
A_dat  <- rnorm(n = 1000000, mean = 0, sd = 0.4);
A_mat  <- matrix(data = A_dat, nrow = 1000);
C_dat  <- rbinom(n = 1000 * 1000, size = 1, prob = 1);
C_mat  <- matrix(data = C_dat, nrow = 1000, ncol = 1000);
A_mat     <- A_mat * C_mat;
gammas <- runif(n = 1000, min = 0, max = 2);
mu_gam <- mean(gammas);
diag(A_mat) <- -1;
```

```
A1      <- gammas * A_mat;
A0      <- mu_gam * A_mat;
A0_e    <- eigen(A0)$values;
A0_r    <- Re(A0_e);
A0_i    <- Im(A0_e);
A1_e    <- eigen(A1)$values;
A1_r    <- Re(A1_e);
A1_i    <- Im(A1_e);

A0_vm        <- A0;
diag(A0_vm) <- NA;
A0vec        <- as.vector(A0_vm);
A0vec        <- A0vec[is.na(A0vec) == FALSE];
A1_vm        <- A1;
diag(A1_vm) <- NA;
A1vec        <- as.vector(A1_vm);
A1vec        <- A1vec[is.na(A1vec) == FALSE];
```

26 The code below reproduces the figure itself.

```
par(mfrow = c(1, 2), mar = c(0.5, 0.5, 0.5, 0.5), oma = c(5, 5, 0, 0));
plot(A0_r, A0_i, xlim = c(-16.5, 15.5), ylim = c(-16.5,15.5), pch = 4, cex = 0.7,
     xlab = "", ylab = "", cex.lab = 1.5, cex.axis = 1.5, asp = 1);
vl <- seq(from = 0, to = 2*pi, by = 0.001);
x0 <- sqrt(1000) * sd(A0vec) * cos(vl) + mean(diag(A0));
y0 <- sqrt(1000) * sd(A0vec) * sin(vl);
x1 <- sqrt(1000) * sd(A1vec) * cos(vl) + mean(diag(A1));
y1 <- sqrt(1000) * sd(A1vec) * sin(vl);
text(x = -15.5, y = 19, labels = "a", cex = 2);
points(x = x0, y = y0, type = "l", lwd = 3);
points(x = x1, y = y1, type = "l", col = "red", lwd = 3, lty = "dashed");
plot(A1_r, A1_i, xlim = c(-16.5, 15.5), ylim = c(-16.5,15.5), pch = 4, cex = 0.7,
     xlab = "", ylab = "", cex.lab = 1.5, cex.axis = 1.5, asp = 1, col = "red",
     yaxt = "n");
text(x = -15.5, y = 19, labels = "b", cex = 2);
points(x = x1, y = y1, type = "l", col = "red", lwd = 3)
points(x = x0, y = y0, type = "l", lwd = 3, lty = "dashed");
mtext(side = 1, "Real", outer = TRUE, line = 3, cex = 2);
mtext(side = 2, "Imaginary", outer = TRUE, line = 2.5, cex = 2);
```



Stability across increasing S

```r
dat <- read.csv(file = "sim_results/C_1/random_all.csv");
dat <- dat[,-1];
```

The table below shows the results for all simulations of random M matrices at $\sigma = 0.4$ and $C = 1$ given a range of S from 2 to 32. In this table, the `A0` refers to matrices when $\gamma = 1$, while `A1` refers to matrices after $Var(\gamma)$ is added and $\gamma \sim \mathcal{U}(0, 2)$. Each row summarises data for a given S over 1 million randomly simulated M (`A0` and `A1`). The column `A0_unstable` shows the number of `A0` matrices that are unstable, and the column `A0_stable` shows the number of `A0` matrices that are stable (these two columns sum to 1 million). Similarly, the column `A1_unstable` shows the number of `A1` matrices that are unstable and `A1_stable` shows the number that are stable. The columns `A1_stabilised` and `A1_destabilised` show how many `A0` matrices were stabilised or destabilised, respectively, by $Var(\gamma)$.

S	A0_unstable	A0_stable	A1_unstable	A1_stable	A1_stabilised	A1_destabilised
2	293	999707	293	999707	0	0
3	3602	996398	3609	996391	0	7
4	14937	985063	15008	984992	0	71
5	39289	960711	39783	960217	36	530
6	78845	921155	80207	919793	389	1751
7	133764	866236	136904	863096	1679	4819
8	204112	795888	208241	791759	5391	9520
9	288041	711959	291775	708225	12619	16353
10	384024	615976	384931	615069	23153	24060
11	485975	514025	481019	518981	35681	30725

S	A0_unstable	A0_stable	A1_unstable	A1_stable	A1_stabilised	A1_destabilised
12	590453	409547	577439	422561	48302	35288
13	689643	310357	669440	330560	57194	36991
14	777496	222504	751433	248567	60959	34896
15	850159	149841	821613	178387	58567	30021
16	905057	94943	877481	122519	51255	23679
17	943192	56808	919536	80464	40854	17198
18	969018	30982	949944	50056	30102	11028
19	984301	15699	970703	29297	20065	6467
20	992601	7399	983507	16493	12587	3493
21	996765	3235	991532	8468	7030	1797
22	998693	1307	995567	4433	3884	758
23	999503	497	997941	2059	1883	321
24	999861	139	999059	941	899	97
25	999964	36	999617	383	380	33
26	999993	7	999878	122	121	6
27	999995	5	999946	54	53	4
28	1000000	0	999975	25	25	0
29	1000000	0	999997	3	3	0
30	1000000	0	999999	1	1	0
31	1000000	0	999999	1	1	0
32	1000000	0	1000000	0	0	0
33	1000000	0	1000000	0	0	0
34	1000000	0	1000000	0	0	0
35	1000000	0	1000000	0	0	0
36	1000000	0	1000000	0	0	0
37	1000000	0	1000000	0	0	0
38	1000000	0	1000000	0	0	0
39	1000000	0	1000000	0	0	0
40	1000000	0	1000000	0	0	0
41	1000000	0	1000000	0	0	0
42	1000000	0	1000000	0	0	0
43	1000000	0	1000000	0	0	0
44	1000000	0	1000000	0	0	0
45	1000000	0	1000000	0	0	0
46	1000000	0	1000000	0	0	0
47	1000000	0	1000000	0	0	0
48	1000000	0	1000000	0	0	0
49	1000000	0	1000000	0	0	0
50	1000000	0	1000000	0	0	0

The results underlying this table were produced with the `rand_gen_var` function below.

```
rand_gen_var <- function(max_sp, iters, int_type = 0, rmx = 0.4, C = 1){
    tot_res <- NULL;
    fea_res <- NULL;
    for(i in 2:max_sp){
        iter           <- iters;
        tot_res[[i-1]] <- matrix(data = 0, nrow = iter, ncol = 7);
        fea_res[[i-1]] <- matrix(data = 0, nrow = iter, ncol = 7);
        while(iter > 0){
            r_vec    <- rnorm(n = i, mean = 0, sd = rmx);
            A0_dat   <- rnorm(n = i * i, mean = 0, sd = 0.4);
```

```
            A0       <- matrix(data = A0_dat, nrow = i, ncol = i);
            A0       <- species_interactions(mat = A0, type = int_type);
            C_dat    <- rbinom(n = i * i, size = 1, prob = C);
            C_mat    <- matrix(data = C_dat, nrow = i, ncol = i);
            A0       <- A0 * C_mat;
            diag(A0) <- -1;
            gam1     <- runif(n = i, min = 0, max = 2);
            A1       <- A0 * gam1;
            A0       <- A0 * mean(gam1);
            A0_stb   <- max(Re(eigen(A0)$values)) < 0;
            A1_stb   <- max(Re(eigen(A1)$values)) < 0;
            A0_fea   <- min(-1*solve(A0) %*% r_vec) > 0;
            A1_fea   <- min(-1*solve(A1) %*% r_vec) > 0;
            if(A0_stb == TRUE){
                tot_res[[i-1]][iter, 1] <- 1;
            }
            if(A1_stb == TRUE){
                tot_res[[i-1]][iter, 2] <- 1;
            }
            if(A0_fea == TRUE){
                fea_res[[i-1]][iter, 1] <- 1;
            }
            if(A1_fea == TRUE){
                fea_res[[i-1]][iter, 2] <- 1;
            }
            iter    <- iter - 1;
        }
        print(i);
    }
    all_res <- summarise_randmat(tot_res = tot_res, fea_res = fea_res);
    return(all_res);
}
```

The above function calls the two functions `species_interactions` and `summarise_randmat`, which are provided below.

```
species_interactions <- function(mat, type = 0){
    if(type == 1){
        mat[mat > 0] <- -1*mat[mat > 0];
    }
    if(type == 2){
        mat[mat < 0] <- -1*mat[mat < 0];
    }
    if(type == 3){
        for(i in 1:dim(mat)[1]){
            for(j in 1:dim(mat)[2]){
                if(mat[i, j] * mat[j, i] > 0){
                    mat[j, i] <- -1 * mat[j, i];
                }
            }
        }
    }
    return(mat);
}
```

```r
summarise_randmat <- function(tot_res, fea_res){
    sims    <- length(tot_res);
    all_res <- matrix(data = 0, nrow = sims, ncol = 13);
    for(i in 1:sims){
        all_res[i, 1]  <- i + 1;
        # Stable and unstable
        all_res[i, 2]  <- sum(tot_res[[i]][,1] == FALSE);
        all_res[i, 3]  <- sum(tot_res[[i]][,1] == TRUE);
        all_res[i, 4]  <- sum(tot_res[[i]][,2] == FALSE);
        all_res[i, 5]  <- sum(tot_res[[i]][,2] == TRUE);
        # Stabilised and destabilised
        all_res[i, 6] <- sum(tot_res[[i]][,1] == FALSE &
                             tot_res[[i]][,2] == TRUE);
        all_res[i, 7] <- sum(tot_res[[i]][,1] == TRUE &
                             tot_res[[i]][,2] == FALSE);
        # Feasible and infeasible
        all_res[i, 8]  <- sum(fea_res[[i]][,1] == FALSE);
        all_res[i, 9]  <- sum(fea_res[[i]][,1] == TRUE);
        all_res[i, 10]  <- sum(fea_res[[i]][,2] == FALSE);
        all_res[i, 11]  <- sum(fea_res[[i]][,2] == TRUE);
        # Feased and defeased
        all_res[i, 12] <- sum(fea_res[[i]][,1] == FALSE &
                             fea_res[[i]][,2] == TRUE);
        all_res[i, 13] <- sum(fea_res[[i]][,1] == TRUE &
                             fea_res[[i]][,2] == FALSE);
    }
    cnames <- c("N", "A0_unstable", "A0_stable", "A1_unstable", "A1_stable",
                "A1_stabilised", "A1_destabilised", "A0_infeasible",
                "A0_feasible", "A1_infeasible", "A1_feasible",
                "A1_made_feasible", "A1_made_infeasible");
    colnames(all_res) <- cnames;
    return(all_res);
}
```

Note that feasibility results were ommited for the table above, but are shown below.