Ionotropic receptors as the driving force behind human synapse establishment

Supplementary Material

Lucas H. Viscardi Danilo O. Imparato Maria Cátira Bortolini
Rodrigo J. S. Dalmolin

Abstract

Model uncertainty and limited data are fundamental challenges to robust management of human intervention in a natural system. These challenges are acutely highlighted by concerns that many ecological systems may contain tipping points, such as Allee population sizes. Before a collapse, we do not know where the tipping points lie, if they exist at all. Hence, we know neither a complete model of the system dynamics nor do we have access to data in some large region of state-space where such a tipping point might exist.

Contents

Project structure

This is the title page

Preprocessing

This topic refers mainly to data wrangling done before the actual analysis with the intent of making it simpler.

Eukaryota species tree

We opted to use the TimeTree database in order to obtain an standardized Eukaryota species tree. However, some species were not present in it, so we devised a way to fill them in based on NCBI Taxonomy data.

NCBI Taxonomy tree

First we preprocess NCBI Taxonomy data to leave only STRING eukaryotes, thus making the task easier.

Downloading data

```
download_if_missing("http://ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdump.tar.gz")
download_if_missing("stringdb-static.org/download/species.v11.0.txt")

untar("download/taxdump.tar.gz", exdir = "download/taxdump")
```

Loading data

Table 1: Lists all organisms in STRING v11.

Location: data-raw/download/species.v11.0.txt Source: stringdb-static.org/download/species.v11.0.txt					
#	Col. name	Col. type	Used?	Example	Description
1	taxid	character	yes	9606	NCBI Taxonomy identifier
2	string_type	character	no	core	if the genome of this species is core or periphery
3	string_name	character	yes	Homo sapiens	STRING species name
4	ncbi_official_name	character	no	Homo sapiens	NCBI Taxonomy species name

Table 2: Links outdated taxon IDs to corresponding new ones.

Location: data-raw/download/taxdump/merged.dmp Source: ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdump.tar.gz					
#	Col. name	Col. type	Used?	Example	Description
1	taxid	character	yes	9606	id of node that has been merged
2	new_taxid	character	yes	core	id of node that is the result of merging

Table 3: Represents taxonomy nodes.

Location: data-raw/download/taxdump/nodes.dmp Source: ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdump.tar.gz					
#	Col. name	Col. type	Used?	Example	Description
1	taxid	character	yes	2	node id in NCBI taxonomy database
2	parent_taxid	character	yes	131567	parent node id in NCBI taxonomy database
3	rank	character	no	superkingdom	rank of this node
4	...		no		(too many unrelated fields)

Table 4: Links taxon IDs to actual species names.

Location: data-raw/download/taxdump/names.dmp Source: ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdump.tar.gz					
#	Col. name	Col. type	Used?	Example	Description
1	taxid	character	yes	2	the id of node associated with this name
2	name	character	yes	Monera	name itself
3	unique_name	character	no	Monera <bacteria>	the unique variant of this name if name not unique
4	name_class	character	yes	scientific name	type of name

```
string_species <- read_tsv(
  "download/species.v11.0.txt",
  skip = 1,
  col_names = c(
    "taxid",
    "string_type",
    "string_name",
    "ncbi_official_name"
  ),
  col_types = cols_only(
    taxid = "c",
    string_name = "c"
  )
)

# these .dmp files are very tricky to read
# the following read_delims are very hacky
ncbi_merged_ids <- read_delim(
  "download/taxdump/merged.dmp",
  delim = "|",
  trim_ws = TRUE,
  col_names = c("taxid","new_taxid"),
  col_types = "cc"
)

ncbi_edgelist <- read_delim(
  "download/taxdump/nodes.dmp",
  skip = 1,
  delim = "|",
  trim_ws = TRUE,
  col_names = c("n1","n2"),
  col_types = "cc"
)

ncbi_taxon_names <- read_delim(
  "download/taxdump/names.dmp",
  delim = "|",
  trim_ws = TRUE,
  col_names = c("name","ncbi_name","type"),
  col_types = "cc-c"
)
```

Updating STRING taxon IDs

Some organisms taxon IDs are outdated in STRING. We must update them to work with the most recent NCBI Taxonomy data.

```
string_species %<>%
  left_join(ncbi_merged_ids) %>%
  mutate(new_taxid = coalesce(new_taxid, taxid))
```

Creating tree graph

The first step is to create a directed graph representing the NCBI Taxonomy tree.

```
# leaving only "scientific name" rows
ncbi_taxon_names %<>%
  filter(type == "scientific name") %>%
  select(name, ncbi_name)

# finding Eukaryota taxid
eukaryota_taxon_id <- subset(ncbi_taxon_names, ncbi_name == "Eukaryota", "name", drop = TRUE)

# creating graph
g <- graph_from_data_frame(ncbi_edgelist[,2:1], directed = TRUE, vertices = ncbi_taxon_names)

# easing memory
rm(ncbi_edgelist, ncbi_merged_ids)
```

Traversing the graph

The second step is to traverse the graph from the Eukaryota root node to STRING species nodes. This automatically drops all non-eukaryotes and results in a species tree representing only STRING eukaryotes (476).

```
eukaryote_root <- V(g)[eukaryota_taxon_id]
eukaryote_leaves <- V(g)[string_species[["new_taxid"]]]

# not_found <- subset(string_species, !new_taxid %in% ncbi_taxon_names$name)

eukaryote_paths <- shortest_paths(g, from = eukaryote_root, to = eukaryote_leaves, mode = "out")$vpath

eukaryote_vertices <- eukaryote_paths %>% unlist %>% unique

eukaryote_tree <- induced_subgraph(g, eukaryote_vertices, impl = "create_from_scratch")
```

Saving

Saving `ncbi_tree` and `string_eukaryotes` for package use. These data files are documented by the package. We also create a plain text file `476_ncbi_eukaryotes.txt` containing the updated names of all 476 STRING eukaryotes. This file will be queried against the TimeTree website.

```
ncbi_tree <- treeio::as.phylo(eukaryote_tree)

# plot(ncbi_tree %>% ape::ladderize(), type="cladogram")

string_eukaryotes <- string_species %>%
  filter(new_taxid %in% ncbi_tree$tip.label) %>%
  inner_join(ncbi_taxon_names, by = c("new_taxid" = "name"))

write(string_eukaryotes[["ncbi_name"]],"476_ncbi_eukaryotes.txt")

# usethis::use_data(ncbi_tree, overwrite = TRUE)
write.tree(ncbi_tree, "ncbi_tree.nwk")
usethis::use_data(string_eukaryotes, overwrite = TRUE)
```

```
## <U+2714> Setting active project to 'C:/R/neuro'
## <U+2714> Saving 'string_eukaryotes' to 'data/string_eukaryotes.rda'
```

Duplicated Genera

Some species from different kingdoms may share the same genus name. These genera must be noted down because one of the ways we fill in missing species is by looking at genera names.

Loading data

See Table 3 and Table 4.

```
taxid_rank <- read_delim(
  "download/taxdump/nodes.dmp",
  skip = 1,
  delim = "|",
  trim_ws = TRUE,
  col_names = c("taxid","rank"),
  col_types = "c-c"
)

ncbi_taxon_names <- read_delim(
  "download/taxdump/names.dmp",
  delim = "|",
  trim_ws = TRUE,
  col_names = c("taxid","ncbi_name","type"),
  col_types = "cc-c"
)
```

Finding duplicated genera

```
# keeping genera nodes
taxid_rank %<>% filter(rank == "genus")

# keeping scientific names
ncbi_taxon_names %<>%
  filter(type == "scientific name") %>%
  select(taxid, ncbi_name) %>%
  inner_join(taxid_rank)

# extracting and saving duplicated values
duplicated_genera <- ncbi_taxon_names %>%
  pull(ncbi_name) %>%
  extract(duplicated(.)) %>%
  write("duplicated_genera.txt")
```

Hybrid tree

Once we have both the NCBI eukaryotes tree and the list of duplicated genera, we can start assembling the complete hybrid tree.

Downloading data

Besides downloading all TimeTree species data (`Eukaryota_species.nwk`) we also need to manually query the website for the 476 STRING eukaryotes (`476_ncbi_eukaryotes.txt`). The file is called `476_ncbi_eukaryotes.txt` because it contains updated NCBI Taxonomy names rather than STRING outdated names. This ensures better results.

```
download_if_missing(
  paste0("http://timetree.org/ajax/direct_download",
         "?direct-download-format=newick",
         "&direct-download-id=23070",
         "&direct-download-rank=species"),
  "Eukaryota_species.nwk"
)
```

Loading data

```
# loading species names and taxon ids
data(string_eukaryotes, package = "neurotransmissionevolution")

# loading newick tree manually obtained from timetree
timetree_newick <- read.tree("download/timetree_335_eukaryotes.nwk")

# the following genera names are unreliable and should not be searched for
duplicated_genera <- scan("duplicated_genera.txt", what = "character")

# loading all TimeTree species data we have just download (85000 species)
tree_85k <- read.tree("download/Eukaryota_species.nwk")
```

Unfound species with matching genera

Some of the 476 STRING eukaryotes are not present in the TimeTree database. However, sometimes TimeTree does contain tree data for closely related species (e.g. *Monosiga brevicollis* is not present, but *Monosiga ovata* is). Therefore, we can use these closely related species as proxies for the actual species. This is done by searching for genera names in the complete database (`Eukaryota_species.nwk`). In the given *Monosiga brevicollis* example, we search for *Monosiga* in the complete database. We see that there is information for at least one other species of the *Monosiga* genus (in this case, *Monosiga ovata*), so we add *Monosiga brevicollis* as a sister branch to the found species.

When you search for a term in TimeTree, it uses a synonym list obtained from NCBI to try to resolve it. Sometimes TimeTree will resolve a searched term to a scientific name different from the one you searched for. The problem with this is that TimeTree does not make it obvious that it is returning a different term. The first step is to find out which species resolved to different names in the `timetree_335_eukaryotes.nwk` file:

```
# plot(timetree_newick %>% ladderize, type = "cladogram", use.edge.length = F)

# replacing timetree species underscores with spaces
timetree_newick[["tip.label"]] %<>% str_replace_all("_", " ")

# which timetree species' names exactly match with ncbi's
taxid_indexes <- timetree_newick[["tip.label"]] %>% match(string_eukaryotes[["ncbi_name"]])

# find out which timetree species names didn't exactly match ncbi's
unmatched_names <- timetree_newick[["tip.label"]] %>% magrittr::extract(taxid_indexes %>% is.na)
print(unmatched_names)
```

```
## [1] "Cercospora fijiensis"     "Arthroderma benhamiae"
## [3] "Macropus eugenii"         "Ostreococcus lucimarinus"
## [5] "Oryza nivara"
```

```
# manually creating lookup table to be joined
ncbi_to_timetree <- tribble(
  ~timetree_name,              ~ncbi_name,
  "Cercospora fijiensis",      "Pseudocercospora fijiensis",
  "Arthroderma benhamiae",     "Trichophyton benhamiae",
  "Macropus eugenii",          "Notamacropus eugenii",
  "Ostreococcus lucimarinus",  "Ostreococcus sp. 'lucimarinus'",
  "Oryza nivara",              "Oryza sativa f. spontanea"
)

# joining info
species_dictionary <- string_eukaryotes %>% left_join(ncbi_to_timetree)

# coalescing NAs to ncbi_name
species_dictionary %<>%
  mutate(timetree_name = coalesce(timetree_name, ncbi_name)) %>%
  mutate(timetree_name = ifelse(timetree_name %in% timetree_newick[["tip.label"]], timetree_name, NA))
```

Now we can start looking for unfound species genera in the complete tree data.

```
# annotating genera
species_dictionary %<>%
  mutate(genus_search = coalesce(timetree_name, ncbi_name) %>%
  strsplit(" ") %>%
  sapply("[", 1))
```

```
# unique genera
selected_genera <- species_dictionary[["genus_search"]] %>% unique

# these are unreliable selected_genera:
unreliable_genera <- intersect(selected_genera, duplicated_genera)

# ensuring a cleaner newick file with only necessary data
# this is actually really important
tree_85k[["node.label"]] <- NULL
tree_85k[["edge.length"]] <- NULL

# replacing timetree's underscores with spaces
tree_85k[["tip.label"]] %<>% str_replace_all("_", " ")

# storing genus
tree_85k[["tip.genus"]] <- sapply(strsplit(tree_85k[["tip.label"]]," "), "[", 1)
tree_85k_genera <- tree_85k[["tip.genus"]] %>% unique

# subtracting unreliable genera
tree_85k_genera %<>% setdiff(unreliable_genera)

# keeping only selected genera, including unreliable ones
tree_genus <- tree_85k %$% keep.tip(., tip.label[tip.genus %in% selected_genera])
tree_genus[["tip.genus"]] <- sapply(strsplit(tree_genus[["tip.label"]]," "), "[", 1)

# unfound species which genera are present in the 85k tree
unfound_species <- species_dictionary %>%
  filter(is.na(timetree_name) & genus_search %in% tree_85k_genera)
```

Once we figured out which species have proxy genera in the complete data, we can start filling them in as sister branches.

```
# for each unfound species which genus is present in the 85k tree,
for(i in 1:nrow(unfound_species)){
  # we search for all species of this genus ("sister species") in the 85k tree
  # this part is tricky because bind.tip rebuilds the tree from scratch
  # so we need to keep removing underscores. there are better ways to do this.
  tip_genus <- tree_genus[["tip.label"]] %>% strsplit("[_ ]") %>% sapply("[", 1)
  sister_species <- tree_genus[["tip.label"]][tip_genus == unfound_species[[i, "genus_search"]]]
  # we obtain the sister_species' most recent common ancestor (MRCA)
  # c(.[1]) is a hack because the MRCA function only works with at least 2 nodes
  where <- getMRCA(tree_genus, sister_species %>% c(.[1]))
  # and then add a leaf node linked to this MRCA
  tree_genus %<>% bind.tip(tip.label = unfound_species[[i, "ncbi_name"]], where = where)
}

# for some reason bind.tip adds underscores to species names
tree_genus[["tip.label"]] %<>% str_replace_all("_", " ")

# keeping track of found species
found_species <- species_dictionary %>% filter(!is.na(timetree_name) | genus_search %in% tree_85k_genera)
# forced_name means it either was found in timetree or we forced it by looking at genera names
found_species %<>% mutate(forced_name = coalesce(timetree_name, ncbi_name))

# so we keep only found species in this tree we are building (timetree + forced by genera)
tree_genus %<>% keep.tip(found_species[["forced_name"]])

# which found_species rows correspond to each tip.label?
match_tiplabel_name <- match(tree_genus[["tip.label"]], found_species[["forced_name"]])

tree_genus %<>% list_modify(
# converting to ncbi taxids
  tip.label = found_species[["new_taxid"]][match_tiplabel_name]
)
```

Species of unfound genera

In this part, we try to fill in the remaining missing species (those which genera were not found in TimeTree) by searching for their closest relatives (according to NCBI Taxonomy) that are present in the current tree. Once we find its two closest relatives, we can add the missing species as a branch from their LCA. This is a conservative approach.

```
# converting ncbi phylo to igraph
graph_ncbi <- read.tree("ncbi_tree.nwk") %>% as.igraph.phylo(directed = TRUE)
```

```r
# converting phylo to igraph
graph_genus <- as.igraph.phylo(tree_genus, directed = TRUE)

# for each species which genus is not in timetree
# we'll look for its two closest species (in the NCBI tree) which are present in the tree_genus we just built
unfound_genera <- species_dictionary %>% filter(is.na(timetree_name) & !genus_search %in% tree_85k_genera)

# this is the igraph equivalent of "phylo_tree$tip.label"
tip_nodes <- V(graph_ncbi)[degree(graph_ncbi, mode = "out") == 0]

# undirected distances between all species nodes
tip_distances <- graph_ncbi %>%
  distances(v = tip_nodes, to = tip_nodes, mode = "all") %>%
  as_tibble(rownames = "from") %>%
  pivot_longer(-from, names_to = "to", values_to = "distance")

# removing self references (zero distances)
tip_distances %<>% filter(distance > 0)

# we only want to search for species of unfound genera
tip_distances %<>% inner_join(unfound_genera %>% select(from = new_taxid))

# we only want to find species already present in the genus_tree
tip_distances %<>% inner_join(found_species %>% select(to = new_taxid))

# we only want the two closest relatives
tip_distances %<>%
  group_by(from) %>%
  top_n(-2, distance) %>% # top 2 smallest distances
  top_n(2, to) # more than 2 species have the same smallest distance, so we get the first ones

# out distance matrix between all nodes in tree, needed to find MRCAs
out_distances <- graph_genus %>% distances(mode = "out")

# for each species of unfound genera,
# we find the MRCA for its two closest relatives
unfound_genera_mrca <- tip_distances %>% group_by(from) %>% summarise(mrca = {
  # which rows have no infinite distances? the last one represents the MRCA
  mrca_row_index <- max(which(rowSums(is.infinite(out_distances[, to])) == 0))
  rownames(out_distances)[mrca_row_index]
})

# adding unfound genera species nodes
graph_genus %<>% add_vertices(nrow(unfound_genera_mrca), color = "red", attr = list(name = unfound_genera_mrca[["from"]]))

# defining unfound genera species edges
# edges_to_add[1] -> edges_to_add[2], edges_to_add[2] -> edges_to_add[3]...
edges_to_add <- V(graph_genus)[unfound_genera_mrca %>% select(mrca, from) %>% t %>% as.vector]$name

# connecting species leafs to the supposed MRCA
graph_genus %<>% add_edges(V(graph_genus)[edges_to_add])

# plotting
# plot(as.undirected(graph_genus), layout = layout_as_tree(graph_genus), vertex.label = NA, vertex.size=2)

# finally converting to phylo format
phylo_graph_genus <- treeio::as.phylo(graph_genus)

# which species_dictionary rows correspond to each tip.label?
match_tiplabel_taxid <- match(phylo_graph_genus[["tip.label"]], species_dictionary[["new_taxid"]])

phylo_graph_genus %<>% list_modify(
  # adding tip.alias (this is not exported with write.tree)
  tip.alias = species_dictionary[["string_name"]][match_tiplabel_taxid],
  # converting back to string ids
  tip.label = species_dictionary[["taxid"]][match_tiplabel_taxid]
)

# ensuring a cleaner newick file with only necessary data
phylo_graph_genus[["node.label"]] <- NULL
phylo_graph_genus[["edge.length"]] <- NULL

# usethis::use_data(phylo_graph_genus, overwrite = TRUE)
# write.tree(phylo_graph_genus, "../data/hybrid_tree.nwk")
```

Ctenophora as sister to all animals

According to TimeTree, Ctenophora remains as a sister group to Cnidaria. We believe the most recent

consensus in literature is to consider them a sister group to all animals. The following code block moves *Mnemiopsis leidyi*, the only ctenophore in our analysis, to the base of the metazoan lineage.

```
# moving ctenophora before porifera
mnemiopsis_taxid <- species_dictionary %>% filter(ncbi_name == "Mnemiopsis leidyi") %>% pull(taxid)
amphimedon_taxid <- species_dictionary %>% filter(ncbi_name == "Amphimedon queenslandica") %>% pull(taxid)

# reordering tip.labels
from_to <- c(
  "400682" = "27923",  # amphimedon to mnemiopsis
  "10228"  = "400682", # trichoplax to amphimedon
  "27923"  = "10228"   # mnemiopsis to trichoplax
)

modified_phylo <- phylo_graph_genus

modified_phylo[["tip.label"]] %<>% recode(!!!from_to)

write.tree(modified_phylo, "../data/hybrid_tree_modified.nwk")
```

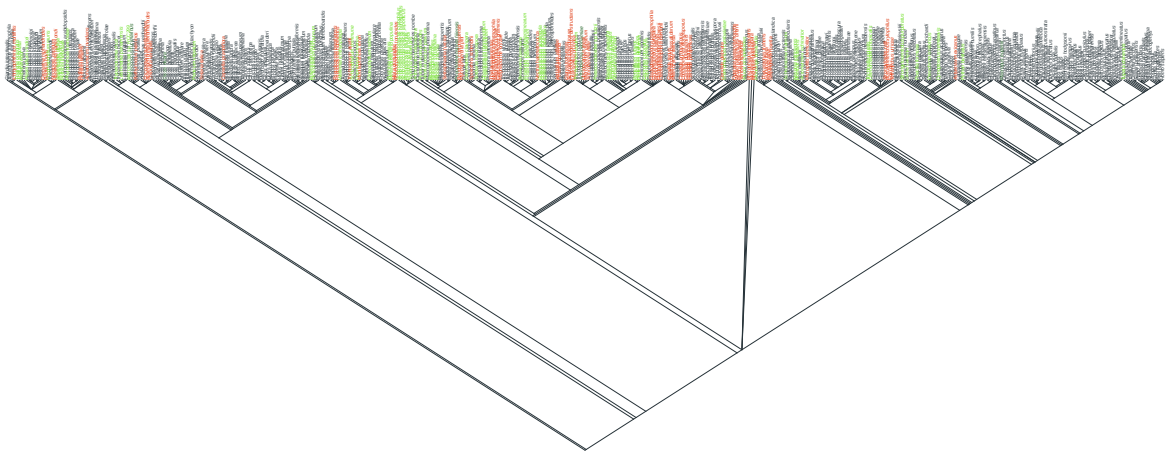

Figure 1: Complete 476 eukaryotes tree. Green species have been filled in by a genus proxy in TimeTree. Red species have been filled in by looking at NCBI Taxonomy.

Gene selection and annotation

Gene selection and annotation

Neuroexclusivity

Explanation

Expression

Pathways

COG data

Network

Analysis

Analysis

```
#leave this chunk
```