Exploit or Explore?
An Empirical Study of Resource Allocation in Scientific Labs

Appendices

A Data and Variable Construction

A.1 Project Rationale

I construct two binary variables, $novel_i$ and $biomedical_i$, based on textual descriptions in the TargetTrack information system. This section describes the variable construction process.

First, I use keywords to identify projects that were novel and/or biomedically important. TargetTrack contains a variable called *targetCategoryList* where labs give projects categorical labels such as "biomedical," "structural coverage,"[29] and so on. It also contains a text field called *targetRationale* where labs give textual descriptions of projects' rationales. Whenever *targetCategoryList* and *targetRationale* contain the following keywords, I set $novel_i$ equal to 1:

big,[30] coverage of protein universe, diversity, first structure of class, low sequence identity, mega,[31] metagenomic, new fold, no structural information, no structure, numer of homologs,[32] pfam, remote homologs, structural coverage, structural template for unsolved, structure coverage, unsolved families, without any solved structures, without structure.

Whenever *targetCategoryList* and *targetRationale* contain the following keywords, I set $biomedical_i$ equal to 1:

activator, adhension, antibiotic, binding, biochemistry, biological interest, biomedical, cascade, catalyze, cell development, community nominated, communit-nominated,[33] community-nominated, community request, conserved, disease, coronavirus, drug, drug development, drug target, effector, enzyme, essential, function, functional studies, functional, gpcr, high value, higvalue,[34] hiv, homeostasis, host, immune, immunity, infection, infectious, inhibitor, interaction, interact, legionella, medical school, metabolism, mitochondria, model system, operon, parkinsons, partnership, pathogen, pathology, pathway, phosphatase, pneumonia, protein family of high biological importance, reagent, receptor, resistance, resistant, salmonella, school of medicine, secret, sensor, shen lab, shen_lab, shen_selection, stem cell, substrate, syndrome, synthesis, t-cell, t cell, therapeutic, thorson lab, toxoplasma, transcription, transport, tuberculosis, tumor, university, vaccine, vibrio, virulence, virulent.

Second, I use labs' selection protocols of projects for additional information. TargetTrack

[29] "Structural coverage" means the project is in part of the structure space with no or few published structures.

[30] BIG and MEGA domain families were defined by the PSI-2 Target Selection Committee as having high value for extensive coverage. These familes contained hundreds to tens of thousands of members and many subfamilies which could not be modeled well due to a lack of structural coverage.

[31] Same as above.

[32] This typo occurs in the raw data.

[33] Same as above.

[34] Same as above.

contains a field where labs describe the protocols they used to conduct each stage of the trials. One type of protocol is the selection protocol. For example, 15 projects were selected because of the protocol "TSel_101," which states "These proteins are important for cell development." I read the descriptions associated with each selection protocol and manually classified whether each protocol was "novel" and/or "biomedical."[35] Then I set $novel_i$ equal to 1 if the project was selected due to a "novel" protocol. I set $biomedical_i$ equal to 1 if the project was selected due to a "biomedical" protocol.

Lastly, TargetTrack has a field that contains a list of reference IDs of each molecule in large-scale bioinformatics databases.[36] These reference ids may yield additional information. Whenever the list of reference ids contains BIG and MEGA reference ids,[37] I set $novel_i$ equal to 1.

A.2 Lab Funding

The NIGMS released the following FOAs directly tied to the PSI.

Table A1: Funding opportunity announcements (FOA) tied to PSI

Id	Title	Year
RFA-GM-99-009	PILOT PROJECTS FOR THE PROTEIN STRUCTURE INITIATIVE (STRUCTURAL GENOMICS)	1999
PA-99-116	PROTEIN STRUCTURE INITIATIVE (STRUCTURAL GENOMICS)	1999
PA-99-117	PROTEIN STRUCTURE INITIATIVE (STRUCTURAL GENOMICS) – SBIR/STTR	1999
RFA-GM-00-006	PILOT PROJECTS FOR THE PROTEIN STRUCTURE INITIATIVE (STRUCTURAL GENOMICS)	2000
RFA-GM-05-001	LARGE-SCALE CENTERS FOR THE PROTEIN STRUCTURE INITIATIVE	2004
RFA-GM-05-002	SPECIALIZED CENTERS FOR THE PROTEIN STRUCTURE INITIATIVE	2004
RFA-GM-06-004	Structural Genomics Knowledgebase (U01)	2006
RFA-GM-10-004	PSI:Biology Knowledgebase (U01)	2009
RFA-GM-10-005	Centers for High-Throughput Structure Determination (U54)	2009
RFA-GM-10-006	Centers for Membrane Protein Structure Determination (U54)	2009
RFA-GM-10-007	Consortia for High-Throughput-Enabled Structural Biology Partnerships (U01)	2009
PAR-10-214	High-Throughput-Enabled Structural Biology Research (U01)	2010
PAR-11-176	High-Throughput-Enabled Structural Biology Partnerships (U01)	2011

[35] The manual classification is available upon request.

[36] These reference ids include, but are not limited to, the molecule's id in the Protein Data Bank (PDB), UniProt, and the National Center for Biotechnology Information (NCBI) database.

[37] See footnote 30.

These FOAs allowed me to search directly all grants associated with them in the NIH RePORT database. In addition to the FOAs, I performed a direct search of the labs' names and abbreviations using RePORT's advanced search functionality to obtain data on each labs' supplementary funding. The search term I used was (quotation marks included):

"[lab full name]" OR "[lab abbreviation]"

I then aggregate each lab's sum of research grants by year from the search results.

A.3 Matching Projects to UniProt Molecule Information

As a preliminary to using the UniProt data, I match projects from the TargetTrack information system to their molecule information on UniProt through two methods.

TargetTrack has a field containing a list of reference ids of each molecule i in large-scale bioinformatics databases. These reference ids include, but are not limited to, the molecule's id in the Protein Data Bank (PDB), UniProt, and the National Center for Biotechnology Information (NCBI) database. When the UniProt id of the molecule is available in this field, the mapping is direct. I also use the following id types, which easily convert into UniProt molecule id through UniProt's ID Mapping service (Huang et al., 2011; UniProt, 2021a):

- *PDB_ID*: a molecule's id in the Protein Data Bank (PDB), a database for 3D structures.
- *P_REFSEQ_AC*: a molecule's id in NCBI's RefSeq protein database.
- *EMBL*: a molecule's corresponding gene's id in European Molecular Biology Laboratory (EMBL)/GenBank/DNA Data Bank of Japan (DDBJ) CDS database.
- *P_ENTREZGENEID*: a molecule's corresponding gene's id in GeneID (Entrez Gene) database.
- *P_GI*: a molecule's GI number assigned by NCBI.

When the first method fails to find a match (usually due to an entirely missing reference id field or obsolete records in the relevant databases), I use a second method: directly searching the molecule's sequence of amino acids against all protein sequences in UniProt.[38] I perform this search using DIAMOND (Buchfink et al., 2015, 2021), a very fast algorithm for searching similar sequences. The diamond command I used was:

```
diamond blastp -d [database name] -q [input sequences in .fasta]
-o [output in .csv] -f 6 qseqid qlen sseqid slen evalue bitscore pident length
-b4.0 --top 5
```

It produces search results with the following variables:

- *qseqid*: query sequence's identifier (the full sequence in this case).
- *qlen*: query sequence's length.
- *sseqid*: search result's UniProt id.

[38] Downloadable in .fasta format at https://www.uniprot.org/downloads.

- *slen*: search result's length.
- *evalue*: the number of expected hits of similar quality that could be found just by chance in a random database of the same size. E-value is a commonly used measure for the degree of similarity between the query sequence and the search result.
- *bitscore*: the required size of a sequence database in which the current match could be found just by chance. Bit score does not depend on the size of the database and is a common alternative measure for the degree of similarity between the query sequence and the search result.
- *pident*: percentage of identical matches between the query sequence and the search result over the alignment length.
- *length*: the alignment length between the query sequence and the search result.

If the query sequence's best match search result, determined by the e-value, a standard metric for assessing sequence similarity, has at least 95% *pident* and the alignment length *length* is at least 67% of both *qlen* and *slen*, I map the query sequence to the result sequence's UniProt id.

I was able to match 262,984 (78.4%) of the 335,553 projects to their UniProt entries through the id mapping method and match an additional 58,593 (17.5%) projects through the direct search. Overall, I was able to map 321,577 (95.8%) projects to their UniProt entries. I then used UniProt's programmatic access for individual entries (UniProt (2021b)) to pull each molecule's information from UniProt. I successfully pulled this information for 319,986 (95.4%) projects.

A.4 Data Glossary

This paper uses hundreds of project characteristics extracted from a variety of sources. This data glossary offers a comprehensive view of these variables.

* Variable is included in the characteristics $\boldsymbol{X}_{ijt}$ in training $\tilde{F}_t(\Omega_t)$.
† Variable is included in the characteristics $\boldsymbol{X}_{ijt}$ in training $F^*(\Omega_T)$.
‡ Variable is included in the characteristics $\boldsymbol{X}_{ijt}$ in training $ridge((\boldsymbol{X}, citation)_T)$.
§ Variable is included in the characteristics $\boldsymbol{X}_{ijt}$ in training $ridge((\boldsymbol{X}, \Delta download)_T)$.
Please see Appendix B for these models.

Table A2: Data glossary

Variables	Description
$[\text{4 cap letters then 6 digits}]_i$*†	Amino acid attributes from the AAindex database (Kawashima et al. (2007)). Each attribute had an identifier that had four capital letters followed by six digits. I started with the 567 attributes in AAindex1, and then normalized and clustered them to a set of around 30 attribute classes as in Babnigg & Joachimiak (2010). I used scikit-learn's implementation of affinity propagation clustering, which automatically picked 34 clusters. I then kept the cluster center of each class. For each cluster center attribute, I calculated the local average value, the local minimum, and the local maximum of the sum of the attribute in a seven-amino acid sliding window for molecule i as in Babnigg & Joachimiak (2010). This resulted in 102 variables.
$[\text{consortium abbreviation}]_{ijt}$*†‡ §	Binary variable = 1 if trial j_i was conducted by the given consortium at time t. Only consortia with more than 70 observations of projects in the TargetTrack database have their corresponding variables. 36 variables in total.
$[\text{gene}]_i$†‡ §	Binary variable = 1 if molecule i is coded for by the given gene. From UniProt. 48,548 variables in total. Most variables are very sparse. For † I only include genes that have occurred more than 200 times in my data. For ‡ and § I only include genes that are associated with at least one molecule whose structure was successfully published in my sample.
$[\text{keyword}]_i$†‡ §	Binary variable = 1 if molecule i is associated with the given keyword in UniProt. Examples of keywords include "Alzheimer disease," "Antioxidant," "RNA-binding," "Viral envelope protein." 1,053 variables in total. Most variables are very sparse. For † I only include keywords that have occurred more than 200 times in my data and remove the keyword "3D-structure" because this is the outcome. For ‡ and § I only include keywords that are associated with at least one molecule whose structure was successfully published in my sample.
$[\text{superkingdom-phylum}]_i$*†‡ §	Binary variable = 1 if molecule i comes from an organisim in the specific superkingdom and phylum. From UniProt. Due to the large number of species molecules in TargetTrack represent, I do not go down the UniProt taxonomy below phylum. 81 variables in total.
$aminoAcid\_[X]_i$*†	Counts the number of times amino acid "X" is in molecule i. 20 variables for each of amino acids A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Contents of certain amino acids are linked to more successes of trials (Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).

$aminoAcidPercent\_[X]_i$*†	Calculate the amino acid "X" content in molecule i in percentages. 20 variables for each of amino acids A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Contents of certain amino acids are linked to more successes of trials (Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).
$biomedical_i$*†‡ §	Binary variable $= 1$ if project i was biomedically important. See Appendix A.1 for variable construction.
$citation_{iy}$	five-year citations and mentions of project i published in year y, from PDBe (Varadi et al. (2020)). When multiple structures were published on molecule i, I take the mean values of the five-year citations and year of publication.
$download_{im}$	Number of downloads of published project i in month m across the three major structure databases in the world, from wwPDB (wwPDB (2013)). Available for Aug 2007–Nov 2013.
$\hat{E}(citation_{iy})$	Expected five-year citations and mentions of project i published in year y. See Appendix B.2 for construction.
$\hat{E}(download_{iy})$	Expected five-year downloads of project i published in year y. See Appendix B.3 for construction.
$\hat{E}_{\tilde{F}_t}(p_{ijt})$	Best-effort replication of the labs' posterior expectation of the probability of success of trial j_i on day t. See Appendix B.1 for construction.
$eukaryote_i$*†‡ §	Maximal percentage identity of molecule i to any eukaryotic molecule. To construct this variable, I search each molecule i against all UniProt protein sequences in the Eukaryota superkingdom (UniProt (2021c)). From the search results, I take the maximal percentage identity of i to any eukaryotic molecule as the variable $eukaryote_i$. Due to potentially large number of search results, the search algorithm DIAMOND (Buchfink et al. (2015, 2021)) by default cuts off results at $evalue = 0.001$. $evalue$ is a well-understood metric for search quality in this field. If there are no search results meeting the cutoff, I let $eukaryote_i = 0$.
$exposedAminoAcid\_[X]_i$*†	Counts the number of times amino acid "X" is on the predicted exposed surface of molecule i. 20 variables for each of amino acids A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y. Exposed surface was predicted using the NetSurfP (Klausen et al. (2019)) program with the cutoff of relative solvent accessibility (rsa) > 0.25. Contents of certain amino acids on the exposed surface of the molecule are linked to more successes of trials (Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).

$exposedAminoAcidPercent\_[X]_i$*†	Calculate the amino acid "X" content on the predicted exposed surface of molecule i in percentages. 20 variables for each of amino acids A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y. Exposed surface was predicted using the NetSurfP (Klausen et al. (2019)) program with the cutoff of relative solvent accessibility (rsa) > 0.25. Contents of certain amino acids on the exposed surface of the molecule are linked to more successes of trials (Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).
$extinctCoeffReduced_i$*†	Molar extinction coefficient of molecule i with reduced cysteines. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Slabinski et al. (2007b) used the extinction coefficient as a feature to predict project success.
$extinctCoeffOxidized_i$*†	Molar extinction coefficient of molecule i with disulfid bridges. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Slabinski et al. (2007b) used the extinction coefficient as a feature to predict project success.
$funding_{ly}$	Total sum of research grants consortium l received from NIH in year y. See Appendix A.2 for variable construction.
$gaps_i$*†	The average number of insertions in molecule i's alignment compared to homologs in UniProt protein sequences. Computed by searching sequence i against UniProt protein sequences using DIAMOND (Buchfink et al. (2015, 2021)). The output variable *gaps* captures this value. Insertions were included as a feature in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Jahandideh et al. (2014).
$gapOpen_i$*†	The average number of insertion openings in the alignment compared to homologs in UniProt protein sequences. Computed by searching sequence i against UniProt protein sequences using DIAMOND (Buchfink et al. (2015, 2021)). The output variable *gapOpen* captures this value. Insertions were included as a feature in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Jahandideh et al. (2014).
$gravyIndex_i$*†	Grand average of hydropathicity index (GRAVY) of molecule i, used to represent the hydrophobicity value of a molecule. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Hydrophobicity is a key determinant of success of trials (Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).
$hasPrevSuccess_{ijkt}$*	Binary variable $= 1$ if at least one previous trial on molecule i successfully completed stage k before date t.
$hasPrevFailure_{ijkt}$*	Binary variable $= 1$ if at least one previous trial on molecule i failed at stage k before date t.

$human_i$*†‡ §	Maximal percentage identity of molecule i to any human molecule. To construct this variable, I search each molecule i against all UniProt protein sequences in the Homo sapiens (human) species (UniProt (2021d)). From the search results, I take the maximal percentage identity of i to any human molecule as the variable $human_i$. Due to potentially large number of search results, the search algorithm DIAMOND (Buchfink et al. (2015, 2021)) by default cuts off results at $evalue = 0.001$. $evalue$ is a well-understood metric for search quality in this field. If there are no search results meeting the cutoff, I let $human_i = 0$.
$instabilityIndex_i$*†	Instability index of molecule i, which is an estimate of the stability of the protein in a test tube. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Instability Index was included as a feature in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Jahandideh et al. (2014).
$isoelectricPoint_i$*†	Isoelectric point of molecule i. Calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module. Isoelectric point is a key determinant of success of trials (Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).
$membrane_i$*†‡ §	Binary variable $= 1$ if project i's UniProt information contains the word "membrane."
$molecularWeight_i$*†‡ §	Molecular weight of molecule i, calculated using Biopython's ProteinAnalysis function from Bio.SeqUtils.ProtParam module.
$novel_i$*†‡ §	Binary variable $= 1$ if project i was novel. See Appendix A.1 for variable construction.
$p^*_{ij,k-1,t_{k-1}}$†	The predicted probability of success of stage $k-1$ of project-trial j_i that started in period t_{k-1}. If $k = 0$, this variable is set to 1.
$percentCoil_i$*†	Predicted percentage of coil secondary structure in molecule i. Predicted using the NetSurfP (Klausen et al. (2019)) program. Secondary structure features were used in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Jahandideh et al. (2014).
$percentCoiledCoil_i$*†	Percentage of coiled-coil regions in molecule i from UniProt. Coiled-coil regions were used in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014).
$percentDisordered_i$*†	Predicted percentage of disordered region in molecule i. Predicted using the NetSurfP (Klausen et al. (2019)) program. Disordered region was used as a feature in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014).

$percentDisorderedUniprot_i$*†	Percentage of disordered region in molecule i from UniProt. Disordered region was used as a feature in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014).
$percentExposed_i$*†	Predicted percentage of amino acids on the exposed surface of molecule i. Exposed surface was predicted using the NetSurfP (Klausen et al. (2019)) program with the cutoff of relative solvent accessibility (rsa) > 0.25. Extent of the exposed surface of the molecule are linked to more successes of trials (Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014)).
$percentHelix_i$*†	Predicted percentage of helix secondary structure in molecule i. Predicted using the NetSurfP (Klausen et al. (2019)) program. Secondary structure features were used in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014).
$percentLowComplexity_i$*†	Predicted percent low-complexity regions in molecule i. Computed using the SEG program (Wootton (1994)). Low-complexity regions were used as features in Slabinski et al. (2007a,b); Jaroszewski et al. (2008).
$percentSignalPeptide_i$*†	Percentage of signal peptide in molecule i. From UniProt. Slabinski et al. (2007a,b); Price et al. (2009a); Babnigg & Joachimiak (2010); Jahandideh et al. (2014) state molecules containing signal peptides have very low chances of success.
$percentStrand_i$*†	Predicted percentage of strand secondary structure in molecule i. Predicted using the NetSurfP (Klausen et al. (2019)) program. Secondary structure features were used in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014).
$percentTransmembraneHelices_i$*†	Percentage of transmembrane helices in molecule i. From UniProt. Transmembrane helices were used as a feature in Slabinski et al. (2007a,b); Jaroszewski et al. (2008); Price et al. (2009a,b); Babnigg & Joachimiak (2010); Jahandideh et al. (2014) .
$pfam_i$	A list of protein families associated with molecule i, from UniProt (UniProt (2021b)).
$phase1_{ijkt}$†	Binary variable $= 1$ if stage k of project-trial j_i started in phase 1 of PSI (pilot phase). I let this variable be 1 if the stage start year is before or in 2005. Phase 1 ended in 2004. However, based on Figures D1 and 6, one can clearly see that 2004 and 2005 are transition periods: output quantity jumped up in 2004. I therefore let 2004 and 2005 be part of both Phase 1 and Phase 2.

$phase2_{ijkt}$†	Binary variable = 1 if stage k of project-trial j_i started in phase 2 of PSI (production phase). I let this variable be 1 if the stage start year is between 2004 and 2010. Phase 2 is between 2005 and 2008. However, based on Figures D1 and 6, one can clearly see that 2009 and 2010 are transition periods: output quantity stayed high but citations reversed the trend. I therefore let 2009 and 2010 be part of both Phase 2 and Phase 3.
$phase3_{ijkt}$†	Binary variable = 1 if stage k of project-trial j_i started in phase 3 of PSI (biomedical phase). I let this variable be 1 if the stage start year is 2009 and beyond.
$prevPub_{iy}$*†‡ §	Number of publications on molecule i by the start of year y, from UniProt (UniProt (2021b)).
$prevStruct_{iy}$*†‡ §	Number of already published structures in the same protein families associated with molecule i by the start of year y. To construct this variable, I first pull from UniProt the list of protein families $pfam_i$ associated with molecule i. I then obtain a mapping of each protein family to its associated structures from EMBL-EBI (2021) and the structures' publication dates (I take the structure's deposition date to the PDB as the publication date) from Varadi et al. (2020). Merging the datasets results in $prevStruct_{iy}$. If i is associated with multiple protein families, I take the average of the number of already published structures in each protein family associated with i.
$prevSuccesses_{ijkt}$*	Number of previous trials on molecule i that have successfully completed stage k before date t.
$prevTrials_{ijkt}$*	Number of previous trials on molecule i that have reached stage k before date t.
$refId_i$	A list of reference ids of molecule i in TargetTrack, used to map i to its information in UniProt.
seq_i	Sequence representation of molecule i's amino acids, unique identifier of project i.
$seqLength_i$*†‡ §	The number of amino acids in molecule i.
$simPrevProj_{it}$	The maximal degree of similarity between project i and all previously attempted projects at time t, measured by the bit score (see Appendix A.3 for the definition of bit score). Computed by searching sequence i against all sequences attempted before time t using DIAMOND (Buchfink et al. (2015, 2021)). The maximum of the output variable $bitscore$ among research results was used as $simPrevProj_{it}$.

$surfaceRuggedness_i$*†	Surface ruggedness of molecule i, defined by the total accessible surface of molecule i divided by the accessible surface predicted based on molecular mass. The total accessible surface of the molecule i is calculated by summing the predicted absolute solvent accessibility of each amino acid from NetSurfP (Klausen et al. (2019)). The accessible surface predicted based on molecular mass is calculated using the formula $6.3(molecularMass)^{0.73}$ (Miller et al. (1987)). Jahandideh et al. (2014) used this variable as a feature.
$trialId_{ij}$	Trial id of project-trial j_i, unique identifier of trial j_i in TargetTrack.
$\widehat{Var}_{\tilde{F}_t}(p_{ijt})$	Best-effort replication of the posterior variance of the labs' beliefs about the probability of success of trial j_i on day t. See Appendix B.1 for construction.
Y_{ijt}	Binary variable $= 1$ if trial j_i on date t was successful.
Y_{ijkt}	Binary variable $= 1$ if intermediate stage k of trial j_i on date t was successful. $Y_{ij0t} = 1$ if DNA was successfully cloned. Y_{ij1t} is only defined when $Y_{ij0t} = 1$ and is equal to 1 if protein was successfully expressed. Y_{ij2t} is only defined when $Y_{ij0t} = 1$ and $Y_{ij1t} = 1$ and is equal to 1 if protein was successfully purified. Y_{ij3t} is only defined when $Y_{ij0t}, Y_{ij1t}, Y_{ij2t} = 1$ and is equal to 1 if protein was successfully crystalized for X-ray crystallography or prepared for NMR or cryo-EM. Y_{ij4t} is only defined when all previous stages were successful and is equal to 1 if the structure was successfully produced and deposited to the Protein Data Bank (PDB) for publication.

B Constructing Posteriors

B.1 Random Forest for Trial Success Probabilities

There are two kinds of models of trial success probability in this paper. The first one is a model that captures how labs formed posterior beliefs. It does not have to produce an unbiased estimate of the true probability of success of a trial. However, it has to produce an unbiased estimate of the labs' perceived posterior beliefs about the probability of success. My implementation of $\tilde{F}_t$ closely follows the machine learning approach the labs described in published journal articles.

The second one is a model of the true data generating process of trial success probability F^*, which is used in simulating counterfactual outcomes. Estimating F^* is different from estimating the posterior using $\tilde{F}_t$ because F^* needs to produce an unbiased estimate of the true probability of success of a trial. As such, my implementation of F^* deviates from $\tilde{F}_t$ in several ways to correct the potential bias of and improve upon the machine learning systems the labs described.

In this appendix, I first explain my implementation of $\tilde{F}_t$; then I move on to discuss how my implementation of F^* deviates from that of $\tilde{F}_t$.

B.1.1 Implementation of $\tilde{F}_t$

My implementation of $\tilde{F}_t$ fits stage-specific models to account for information embodied in outcomes of intermediate stages. Recall that each trial j_i has multiple sequential stages and the overall probability of success of j_i is equal to the product of the probabilities of success for all sequential stages $p_{ijt} = \prod_{k=0}^{4} p_{ijkt}$. The intermediate outcomes Y_{ijk} for stages $k = 0, 1, ...$ up to the point when the overall trial failed/succeeded provides information for future trials' potential.

For a given quarter $q(t)$ and each of the stages $k = 0, 1, 2, 3, 4$, I let the information set $\Omega_{k,q(t)}$ consist of project-trial outcomes realized before quarter $q(t)$ at stage k and these project-trials' characteristics. Following when the labs started to use machine learning to form posterior, I let $q(t)$ to be between 2005 and 2015. For $q(t) = 2005Q1$, I use the trial outcomes realized before 2005 and these trials' characteristics as the initial information set Ω_{2005Q1}. Project-trial characteristics $\boldsymbol{X}_{ijkt}$ I use for training $\tilde{F}_t$ and prediction of labs' posterior beliefs fall under three categories:[39]

- Physicochemical properties of molecule i based on scientific reasoning. These variables were identified by the series of journal articles the labs published and were quite similar across labs and time.
- Other characteristics of project i, for examples, novelty, biomedical importance, and the number of prior publications on molecule i.
- Past successes and failures of project i at stage k.

Then, for the given quarter $q(t)$ and each of the stages $k = 0, 1, 2, 3, 4$, I fit a random forest model $\tilde{F}_{k,q(t)}(\Omega_{k,q(t)})$ using `RandomForestClassifier` from python package `scikit-learn`. Random forest is an *ensemble*[40] machine learning method. The algorithm constructs a large number of decision trees at training time. Each decision tree is a learning model that aims to find the project-trial characteristics predictive of success/failure in the training set. When it comes to prediction, the trained random forest classifier $\tilde{F}_{k,q(t)}(\Omega_{k,q(t)})$ would pool individual trees and average predicted values of $\{\hat{p}_{ijkt}^{ntree}\}$ from individual trees as the final output. As Jahandideh et al. (2014) did, I set the number of trees in the random forest equal to 1000.

Decision trees and random forests are known for often overfitting without regularization. To avoid overfitting, I regularize by restricting the hyperparameters *max_depth*,[41] *min_samples_leaf*,[42] *max_features*,[43] and *min_samples_split*.[44] I perform model selection with a grid search of the

[39] Please see Appendix A.4 for the full list of variables used.

[40] Ensemble methods use multiple learning models to obtain better predictive performance than could be obtained from any of the constituent learning models alone.

[41] This hyperparameter determines the maximum depth of each decision tree.

[42] This hyperparameter determines the minimum number of observations a node in the decision tree must have before it can be split.

[43] This hyperparameter determines the maximum number of features to consider when looking for the best split.

[44] This hyperparameter determines the minimum number of observations required to split a node.

combinations of the four hyperparameters.[45] For each hyperparameter combination, I evaluate the model with five-fold cross validation using `scikit-learn`'s `cross_validate` function. In each iteration of the cross-validation, the function fits a random forest on four out of five cross-validation folds and then computes the cross-validation score by comparing the model's predictions with the actual data from the remaining fold. I use the average log likelihood (`log_loss` scoring in `scikit-learn`) as the cross-validation scoring method. I choose the hyperparameter combination that maximizes the average log likelihood in cross-validation.

After training the models $\tilde{F}_{k,q(t)}$ for $k = 0,1,2,3,4$ for a given $q(t)$, I predict $\widehat{E}_{\tilde{F}_{q(t)}}(p_{ijt})$ and $\widehat{Var}_{\tilde{F}_{q(t)}}(p_{ijt})$ for each project-trial in the choice set C_{lt} at decision time t as follows. I first collect the predictions $\{\hat{p}^{ntree}_{ijkt}\}$ from the 1000 individual decision trees in $\tilde{F}_{k,q(t)}(\Omega_{k,q(t)})$, and then compute $\hat{p}^{ntree}_{ijt} = \prod_{k=0}^{4} \hat{p}^{ntree}_{ijkt}$. There are 1000 values in the set $\{\hat{p}^{ntree}_{ijt}\}$. I let

$$\widehat{E}_{\tilde{F}_{q(t)}}(p_{ijt}) = \bar{p}^{ntree}_{success,ijt}, \tag{14}$$

$$\widehat{Var}_{\tilde{F}_{q(t)}}(p_{ijt}) = s^2(p^{ntree}_{success,ijt}) \tag{15}$$

Although I extensively reference the labs' implementations of machine learning systems when I implement $\tilde{F}_t$, my estimate of the posterior is not a perfect replica of the labs' posteriors. I note why replicating perfectly the labs' posterior beliefs would be difficult and where my implementation corresponds to and deviates from the labs' learning and updating process below:

- I include in $\boldsymbol{X}_{ijt}$ the set of physicochemical properties of molecules identified in the labs' published journal articles (Slabinski et al., 2007a,b; Jaroszewski et al., 2008; Price et al., 2009a,b; Babnigg & Joachimiak, 2010; Jahandideh et al., 2014). This set is the union of the sets of such properties in different articles (to minimize the risk of selection on unobservables) and is fixed for all labs and time periods in my implementation. In contrast, though similar across lab and time, the set of physicochemical properties the labs used in training and prediction still varied. It is impossible to capture all of these potential variations during the labs' long operational history (some may not have been recorded by the published articles).

- The construction of some variables in $\boldsymbol{X}_{ijt}$ requires using software packages that are constantly being updated or have become obsolete. I make my best effort to construct variables using methods as close to the labs' original approach as possible (see Appendix A.4).

- The $\boldsymbol{X}_{ijt}$ in my implementation includes past trial outcomes of projects while the labs' implementations did not explicitly include those characteristics. Still, it is reasonable to believe that researchers working on a project would update their beliefs on the potential of the project upon seeing a trial success/failure.

[45] To reduce computational burden, I do not perform model selection for all $\tilde{F}_{k,q(t)}$. Rather, for each $k = 0,...,4$, I construct $\Omega_{k,T}$ using all outcomes at stage k and only perform model selection for $\tilde{F}_{k,T}$ on this full training set. I then use the selected hyperparameters to train the models $\tilde{F}_{k,q(t)}$ where $q(t) = 2005Q1, 2005Q2, ..., 2015Q4$. The set of *max_depth* used in grid search is $[int(log(sample\_size,2)), 2 \cdot int(log(sample\_size,2)), 3 \cdot int(log(sample\_size,2)), 4 \cdot int(log(sample\_size,2))]$. The set of *min_samples_leaf* used in grid search is $[1,2,4]$. The set of *max_features* used in grid search is $[0.1, 0.2, 0.3, 0.4]$ of the total number of features. The set of *min_samples_split* used in grid search is $[8, 16, 32, 64, 128]$.

- I use random forest as the model of posterior updating for all labs and time periods. In contrast, the machine learning models the labs used in training and prediction varied across lab and across time. It is impossible to capture all of these potential variations during the labs' long operational history (some may not have been recorded by the published articles).

- I set the frequency of "updating" and refitting models at the quarterly interval. In contrast, the labs' actual belief updating frequency is not clearly documented. I use the quarterly interval because training models at a finer interval, such as at the daily frequency, places large computational and storage burden. The day-to-day change of the information set Ω was also relatively small. Therefore, to improve computational tractability, I coarsen the frequency of refitting new models to quarterly.

- My model predicts the overall potential of success of a trial while the labs' implementations focused on predicting the potential of success of bottleneck stages of a trial. That is, for stages where success rates were usually reasonable (for example cloning the DNA), the labs were often not explicitly reliant on something as rigorous as supervised machine learning systems to form and update beliefs, while they were explicitly reliant on such systems for predicting the potential of success in crystallizing a molecule and studying its structure through X-ray crystallography.

- The output the labs' systems produced may not exactly be $\widehat{E}_{\tilde{F}_{q(t)}}(p_{ijt})$ and $\widehat{Var}_{\tilde{F}_{q(t)}}(p_{ijt})$. For example, the model in Slabinski et al. (2007b) predicted the probability of success as an intermediate outcome. The final output was an integer score between 1 and 5, where 1 represents "optimal" and 5 represents "very difficult." The labs' systems did not always predict $\widehat{Var}_{\tilde{F}_{q(t)}}(p_{ijt})$. When they did, the measure took the form of comparing predictions from multiple models side by side (Slabinski et al., 2007a,b; Babnigg & Joachimiak, 2010; Jahandideh et al., 2014). It is reasonable to believe that labs had some understanding that predictions from different models (or submodels of an ensemble model) differed, and looking at how those predictions varied was valuable, though they did not percolate the idea down to form an additional metric just to measure that variation. This seems consistent with the notation that the labs used heuristics to guide their exploration of high-variance projects.

B.1.2 Implementation of F^*

The implementation of F^* is almost identical to that of $\tilde{F}_t$ except for a few deviations. First of all, a new model $\tilde{F}_{k,q(t)}$ (for stages $k = 0,..,4$) is trained for every quarter $q(t)$ between 2005Q1 and 2015Q4, incorporating new trial outcomes realized in each quarter. In contrast, F^*_k (for stages $k = 0,...,4$) is trained only on the full information set Ω_T. Ω_T covers the characteristics and outcomes of all trials in my trial allocations and outcomes dataset in the entire sample period.

Second, F^* uses additional covariates to correct the potential bias of $\tilde{F}_t$ in predicting trial success probabilities. The model $\tilde{F}_t$ may be biased in predicting trial success probabilities because it does not account for the propensity of observing a specific stage of a trial. To see this, think about the probability of success of stage 1 of a trial. We observe stage 1 of a trial only if stage 0 of the trial was successful. If the probabilities of success of stages 0 and 1 are positively correlated, then we are more likely to observe stage 1 of trials that are more likely to succeed in stage 1. Therefore,

models trained with the observed data on stage 1 would produce prediction results that are positively biased. Correcting this bias is simple if we assume that the selection into observing a given stage is only based on observable characteristics of trials: we can use the predicted probability of success of the previous stage as the propensity score of observing the given stage. As such, I include $p^*_{ij,k-1,t_{k-1}}$, the predicted probability of success of stage $k-1$ of trial j_i that started in period t_{k-1}, as a covariate when I train F^*_k. For stage $k=0$, I set this variable to 1. The labs' published articles offer no discussion about this source of bias, so I do not include this variable in training $\tilde{F}_t$.

Another difference between F^* and $\tilde{F}_t$ is that F^* does not include variables on previous trial outcomes as covariates. In simulations, all previous trial outcomes of a project are simulated and should not shift the project's true probability of success; therefore, the simulated counterfactual outcome of a trial should not be based on the simulated previous outcomes.

To further improve the predictive power of F^*, I include in F^* project-trial characteristics the labs did not use in their machine learning systems. I include keywords and genes associated with the molecule i. I also include three phase-specific dummy variables to capture the effects of different phases of the grant program on the probabilities of success, as I learned during my conversations with NIH program officers that the labs underwent retooling corresponding to the changes of phases.

B.2 Ridge Regression for Citations

Let the model be $ridge(\boldsymbol{X}, citation)_T)$, where the training set $(\boldsymbol{X}, citation)_T$ represents the characteristics and $citation_{iy}$ of all published projects in my data. The goal of this model is to predict $E(citation_{iy}|\boldsymbol{X}_{iy}, ridge((\boldsymbol{X}, citation)_T))$, the expected number of five-year citations a project i published in year y would generate conditional on the project's characteristics $\boldsymbol{X}_{iy}$.

The number of characteristics that could potentially predict higher citations is very large. Characteristics ranging from the organism the molecule i is from to the gene that expresses molecule i could all contribute to the biomedical significance and research interests on molecule i. The number of characteristics is on the order of hundreds, most of which are very sparse, while I only have 10,424 observations.[46] This calls for regularization to avoid overfitting.

I use a ridge regression from the python package `scikit-learn`. The project characteristics $\boldsymbol{X}_{iy}$ included for model fitting are shown in Appendix A.4. I choose the regularization hyperparameters using cross-validation with the `RidgeCV` function provided by the `scikit-learn` package.

I standardize the outcome variable $citation_{iy}$ by subtracting away the lab mean value of this variable and then dividing by the lab standard deviation as publications from different labs had large variations in citation numbers. When the model makes a prediction, I multiply the predicted value with the lab standard deviation and add the lab mean to get the predicted citations.

I assess model fit by comparing the actual citations with their out-of-sample predicted citations under five-fold cross validation. Figure B1 shows the distribution of the predicted citations correctly captures a high proportion of zero values in the actual data. Figure B2 shows a scatterplot of predicted citations against the actual citations. A linear regression of the predicted citations on the actual citations without constant shows an $R^2 = 0.580$.

The predicted citations $\hat{E}(citation_{iy}|\boldsymbol{X}_{iy}, Ridge((\boldsymbol{X}, citation)_T))$ has a y subscript because some

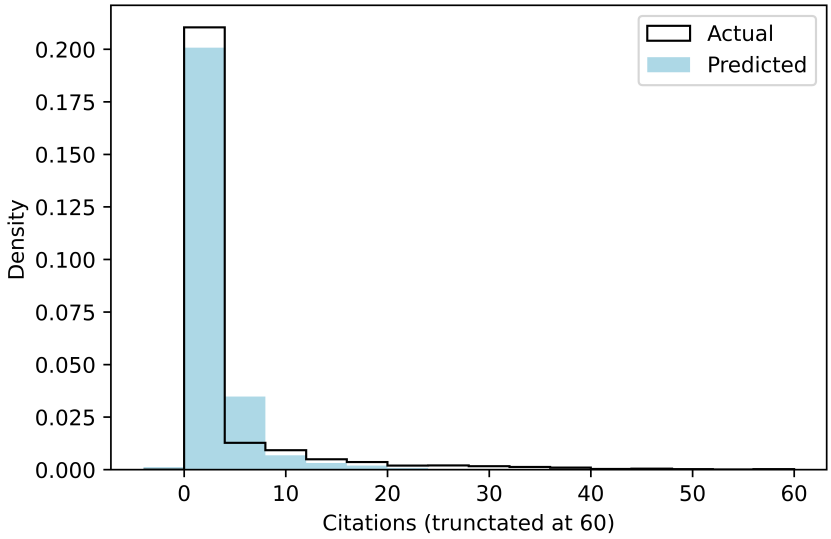
[46] The total number of published projects in my data is 10,501. 77 published projects in my data did not give the PDB ids of their publications so I was not able to map their citation information.

of the important characteristics vary with time, for example, the number of publications on molecule i prior to the year of publication of the structure. In Section 3.3, since we do not know which exact year the structure of each trial in the choice set would have been published if the trial was allocated, I remove the y subscript by averaging the predicted citations for each molecule across years so that

$$\hat{E}(citation_i) = \frac{1}{16} \sum_{y=2000}^{2015} \hat{E}(citation_{iy} | \boldsymbol{X}_{iy}, ridge((\boldsymbol{X}, citation)_T)). \tag{16}$$

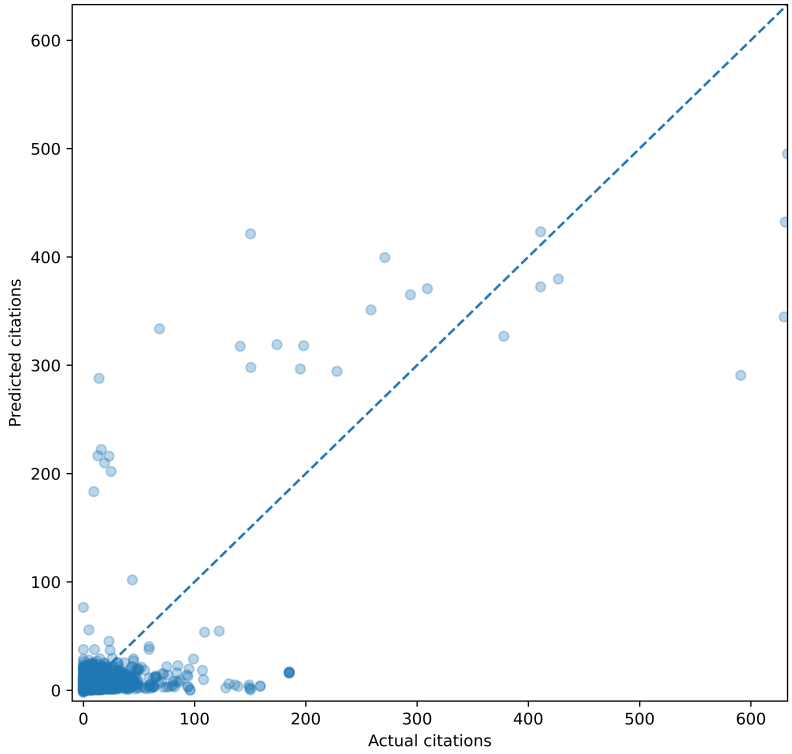
In simulations, I simulate both the outcome (success or failure) of an allocated trial and the date that outcome is realized. In that case, the y subscript is preserved for the predicted citations of the publication.

Figure B1: Distributions of actual citations and predicted citations



Note: Distributions are truncated at 60. Bin width is 4.

Figure B2: Scatterplot of predicted citations against actual citations



B.3 Ridge Regression for Downloads

Before we start predicting downloads, the raw download data needs to be transformed because the number of downloads shows a strong time trend over the lifecycle of a publication. The raw download data I obtain consists of monthly downloads on the structure level between Aug 2007 and Nov 2013 for all publicly available structures human beings know of. The number of structure-month observations is 5,484,800. As some structures in a given month observed were published a long time ago while some just got published, comparing these structures' raw download counts in a month would be misleading. As shown in Figure B3, downloads peak within a month since the publication of a structure and then sharply decline over the following months until reaching some steady level in approximately two years.

I perform a transformation of the raw download data to detrend it as follows. Let $pubAge(i,m)$ be the age of publication i (in months) in month m. To detrend, I first compute $\overline{download}_{pubAge(i,m)}$, the mean downloads of structures that have been published for $pubAge(i,m)$ months.[47] I then compute how much the number of downloads structure i had in month m deviates from this mean, $\widetilde{download}_{im} = download_{im} - \overline{download}_{pubAge(i,m)}$. I then define $\Delta download_i$, the average deviation of structure i's monthly downloads from the mean download trend, by the average of $\widetilde{download}_{im}$, in other words $\Delta download_i = \overline{\widetilde{download}_{im}}$. I treat the variable $\Delta download_i$ as the outcome variable. If there are multiple structures on the same project i, I take the mean of their

[47] I pool observations with $pubAge > 25$ months in computing this mean as the mean number of downloads flattens by 25 months since publication.

average deviations as $\Delta download_i$. I then match the download data with the project-trial characteristics of completed projects in my data.

Let the model be $ridge((\boldsymbol{X}, \Delta download)_T)$, where the training set $(\boldsymbol{X}, \Delta download)_T$ represents the characteristics and $\Delta download_i$ of all published projects in my data. The goal of this model is to predict $E(\Delta download_{iy} | \boldsymbol{X}_{iy}, ridge((\boldsymbol{X}, \Delta download)_T))$, the expected average deviation of the structure's monthly downloads from the mean download trend for a project i published in year y, conditional on the project's characteristics $\boldsymbol{X}_{iy}$. There is a y subscript because some of the important characteristics vary with time, for example, the number of publications on molecule i prior to the year of publication of the structure.

The number of characteristics that could potentially predict higher downloads is very large. Characteristics ranging from those of molecule i's organism to those of molecule i's gene could all contribute to the level of interest on molecule i. The number of characteristics is on the order of hundreds, most of which are very sparse, while I only have 10,424 observations. This calls for regularization to avoid overfitting.

I use a ridge regression from the python package `scikit-learn`. The project characteristics $\boldsymbol{X}_{iy}$ included for model fitting are shown in Appendix A.4. I choose the regularization hyperparameters using cross-validation with the `RidgeCV` function provided by the `scikit-learn` package.

I assess model fit by comparing the actual $\Delta download_{iy}$ with their out-of-sample predicted values under five-fold cross validation. Figure B4 shows a comparison of the distributions. As ridge regression shrinks all regression coefficients towards zero, the distribution of the predicted values is narrower. Still, the predicted values capture the rank order of the actual data well. Figure B5 shows a scatterplot. The plot shows a relationship quite close to the line $y = x$. Figure B6 shows a binned scatterplot.

In Section 3.3 where we show descriptives, I perform additional transformations on the predicted value $\hat{E}(\Delta download_{iy} | \boldsymbol{X}_{iy}, ridge((\boldsymbol{X}, \Delta download)_T))$. First, the predicted value has a y subscript because some of the important characteristics vary with time, for example, the number of publications on molecule i prior to the year of publication of the structure. Since we do not know which exact year the structure of each trial in the choice set would have been published if the trial was allocated, I remove the y subscript by averaging the predicted values for each molecule across years so that

$$\hat{E}(\Delta download_i) = \frac{1}{16} \sum_{y=2000}^{2015} \hat{E}(\Delta download_{iy} | \boldsymbol{X}_{iy}, ridge((\boldsymbol{X}, \Delta download)_T)). \tag{17}$$

Second, the variable $\hat{E}(\Delta download_i)$ predicts the average deviation of monthly downloads from a trend and is difficult to interpret. I therefore transform this variable to a prediction of five-year downloads $\hat{E}(download_i)$ by using the following formula:

$$\hat{E}(download_i) = \sum_{pubAge=0}^{59} \overline{download_{pubAge}} + 60 \times \hat{E}(\Delta download_i), \tag{18}$$

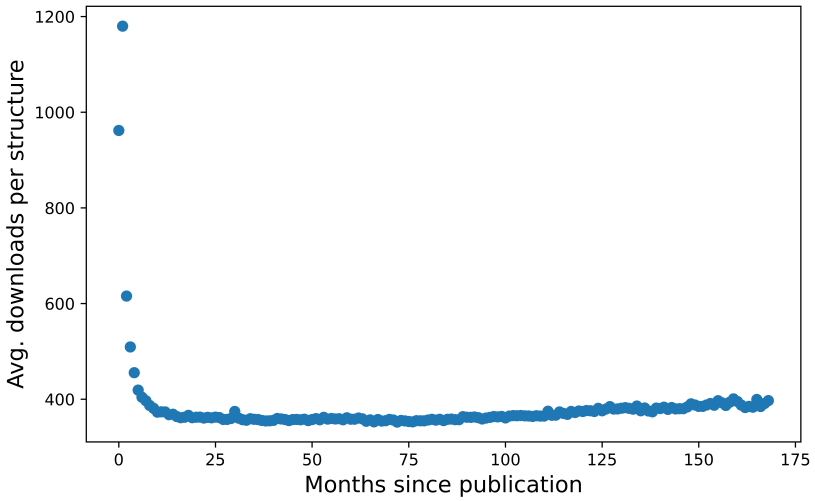
where $\sum_{pubAge=0}^{59} \overline{download_{pubAge}} = 23960.11$ is computed based on the full download data with

5,484,800 structure-month observations.

In simulations, I simulate both the outcome (success or failure) of an allocated trial and the date that outcome is realized. In that case, the y subscript is preserved for the predicted average deviation of monthly downloads of the published projects. Therefore the predicted five-year downloads becomes

$$\hat{E}(download_{iy}) = \sum_{pubAge=0}^{59} \overline{download_{pubAge}} + 60 \times \hat{E}(\Delta download_{iy} | \boldsymbol{X}_{iy}, ridge((\boldsymbol{X}, \Delta download)_T)). \tag{19}$$

Figure B3: Average downloads per structure in months since publication



Note: The plot is based on 5,484,800 structure-month observations of download counts between Aug 2007 and Nov 2013. Each blue dot aggregates in this data the average monthly downloads for structures published m months ago.

Figure B4: Distributions of predicted $\hat{E}(\Delta download_{iy})$ against actual $\Delta download_{iy}$

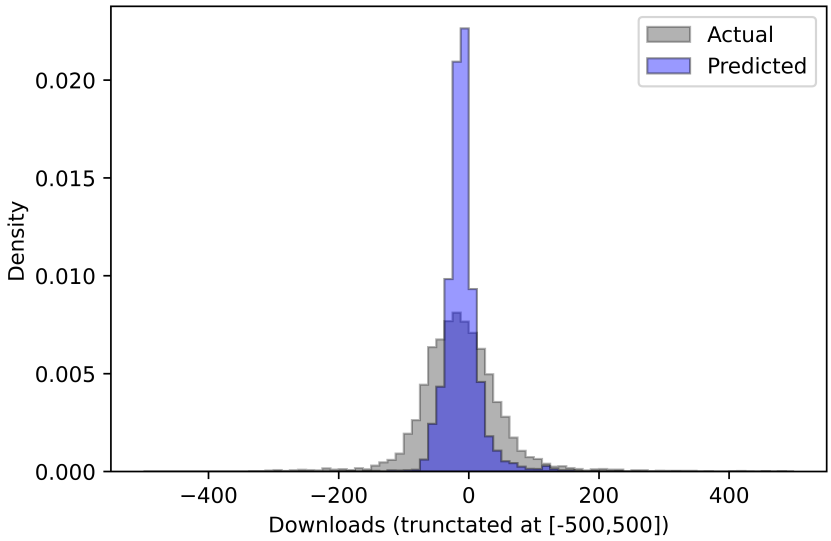


Figure B5: Scatterplot of predicted $\hat{E}(\Delta download_{iy})$ against actual $\Delta download_{iy}$

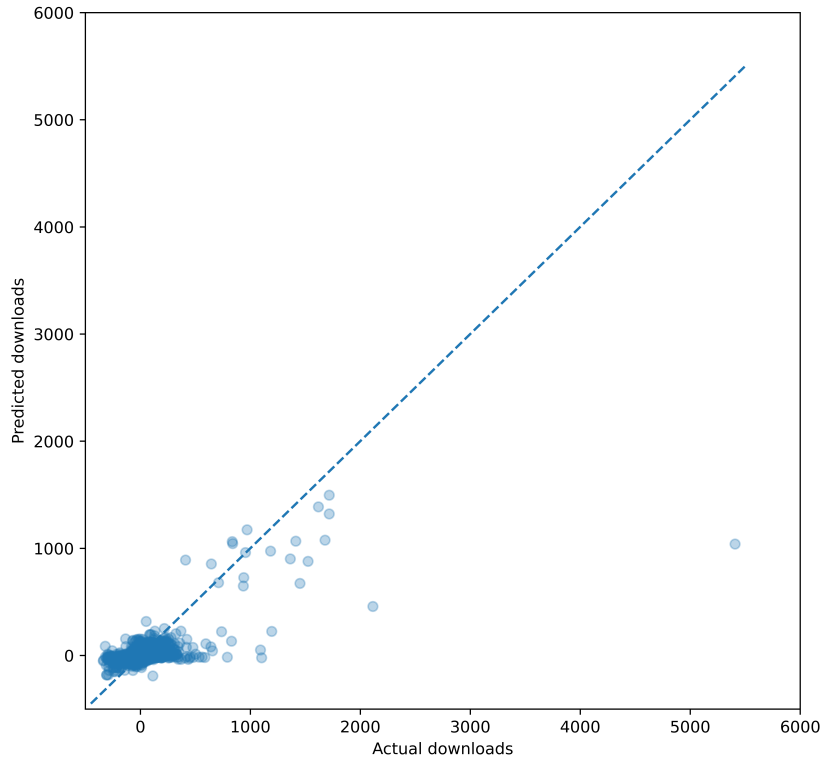
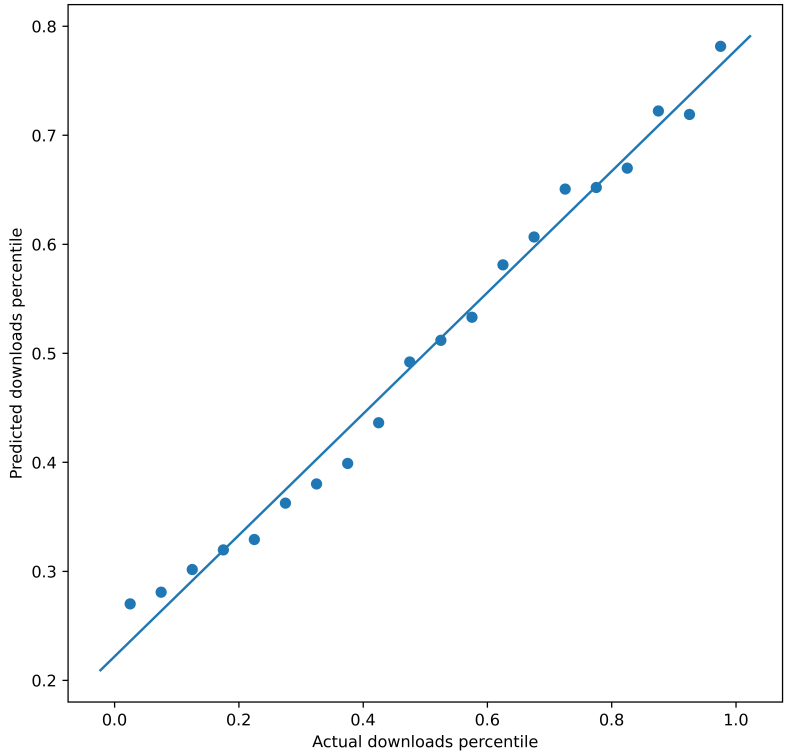


Figure B6: Binned scatterplot of predicted $\hat{E}(\Delta download_{iy})$ against actual $\Delta download_{iy}$



C Additional Details on Modeling and Estimation Procedure

C.1 Functional Form of Payoff Function $\pi_{ijt}(\boldsymbol{a}_{lt}, p_{ijt}; \boldsymbol{\theta}_{Xl})$

To begin, recall that the payoff $\pi_{ijt}(\boldsymbol{a}_{lt}, p_{ijt}; \boldsymbol{\theta}_{Xl})$ of project-trial j_i at t given action $\boldsymbol{a}_{lt}$ has a probability distribution depending on p_{ijt}. The lab or the economist does not perfectly know p_{ijt}, but previous outcomes of trials reveal information about it so one can form a posterior $\tilde{F}_t(p_{ijt}|\Omega_t)$. Integrating $\pi_{ijt}(\boldsymbol{a}_{lt}, p_{ijt}; \boldsymbol{\theta}_{Xl})$ over the posterior, we obtain the posterior expected payoff $\pi_{ijt}(\Omega_t, \boldsymbol{a}_{lt}; \boldsymbol{\theta}_{Xl})$.

I define a function $q(\boldsymbol{a}_{lt}, p_{ijt})$ that maps the probability of success to the probability of *payoff* of project-trial j_i at t given actions. When a project-trial is not allocated on day t, it does not pay off even though it may have a nonzero probability of success. Moreover, the labs often simultaneously allocated multiple trials to the same project. A successful trial j_i of project i should only receive payoff if the simultaneous trials $(j-m)_i, (j-m+1)_i, ..., (j-1)_i$ fail, because we have assumed only the first successful trial/publication on a project produces welfare. Each trial j_i is a Bernoulli trial with probability p_{ijt}. We can express $q(\boldsymbol{a}_{lt}, p_{ijt})$ as follows:

$$q(\boldsymbol{a}_{lt}, p_{ijt}) = a_{ijt}(1-p_{ijt})^m p_{ijt}, \tag{20}$$

where trials $(j-m)_i, (j-m+1)_i, ..., (j-1)_i$ are in the choice set C_{lt} and trial $(j-m)_i$ is the

smallest-numbered trial of project i in C_{lt}.[48] When $m = 0$, trial j_i is the smallest-numbered trial of project i in C_{lt} and its probability of payoff is simply $a_{ijt}p_{ijt}$.

I then specify a deterministic reward function, which captures the amount of payoff a lab will get when a trial pays off. I let the reward function $r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl})$ be a function of project i's characteristics $\boldsymbol{X}_{it}$ on day t. $\boldsymbol{\theta}_{Xl}$ are the welfare weights on $\boldsymbol{X}_{it}$ and are to be estimated. To reduce the number of parameters, I restrict $\boldsymbol{X}_{it}$ to correspond to the set of NIH evaluation metrics. $\boldsymbol{X}_{it}$ includes a constant 1 to capture preference for quantity; $novel_i$ and $prevStruct_{iy}$ to capture preference for novelty; $biomed_i$ and $prevPub_{iy}$ to capture preference for biomedical importance; and $human_i$, $eukaryote_i$, and $membrane_i$ to capture preferences for human, eukaryotic, and membrane proteins, respectively. I let $r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl})$ have a simple linear form

$$r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl}) = 1\cdot\theta_{quant,l} + biomed_i\cdot\theta_{biomed,l} + ... + membrane_i\cdot\theta_{membrane,l}. \tag{21}$$

Whenever a trial pays off, the lab receives a baseline amount $\theta_{quant,l}$ plus additional amounts depending on the other characteristics of the project.

We can then break the posterior expected payoff into a few pieces:

$$\begin{aligned}\int \pi_{ijt}(\boldsymbol{a}_{lt}, p_{ijt};\boldsymbol{\theta}_{Xl})\, d\tilde{F}_t(p_{ijt}|\Omega_t) &= \int r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl})\cdot q(\boldsymbol{a}_{lt}, p_{ijt})\, d\tilde{F}_t(p_{ijt}|\Omega_t)\\ &= a_{ijt}\cdot r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl}) \underbrace{\int \overbrace{[(1-p_{ijt})^m p_{ijt}]}^{\text{let it be } M_{ijt}}\, d\tilde{F}_t(p_{ijt}|\Omega_t)}_{\text{estimated offline}}.\end{aligned} \tag{22}$$

Notice that M_{ijt} only depends on p_{ijt}. Since we have estimated $\tilde{F}_t(p_{ijt}|\Omega_t)$ offline (see Appendix B.1), we can estimate $E_{\tilde{F}_t}(M_{ijt})$ and $Var_{\tilde{F}_t}(M_{ijt})$ offline as well.[49]

Also notice that given the breakdown of the posterior expected payoff in equation (22), V^A_{ijt} does not depend on the full action vector $\boldsymbol{a}_{lt}$. It only depends on the action a_{ijt}. Plugging in equation (22) into V^A_{ijt} in equation (3) and evaluating it at $a_{ijt} = 1$, we obtain:

$$\begin{aligned}V^A_{ijt}(\Omega_t, a_{ijt}=1;\boldsymbol{\theta}_l) &= \int \pi_{ijt}(a_{ijt}=1, p_{ijt};\boldsymbol{\theta}_{Xl})\, d\tilde{F}_t(p_{ijt}|\Omega_t) \;+\; B_{ijt}(\Omega_t, a_{ijt}=1;\boldsymbol{\theta}_{Bl})\\ &= r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl})E_{\tilde{F}_t}(M_{ijt}) + B_{ijt}(\Omega_t, a_{ijt}=1;\boldsymbol{\theta}_{Bl}).\end{aligned} \tag{25}$$

For the main model, $V^A_{ijt}(\Omega_t, a_{ijt}=1;\boldsymbol{\theta}_l) = r(\boldsymbol{X}_{it};\boldsymbol{\theta}_{Xl})E_{\tilde{F}_t}(M_{ijt}) + \sqrt{\frac{\theta_{B1,l}}{j}} + \theta_{B2,l}\cdot(t - t'_{i,t})$. For

[48] $q(\boldsymbol{a}_{lt}, p_{ijt}) = a_{ijt}(1-p_{i,j-m,t})...(1-p_{i,j-1,t})p_{ijt}$. As all trials on day t share the same information set Ω_t, the posteriors for $p_{i,j-m,t}, ..., p_{ijt}$ are the same.

[49] Each trial j_i is a Bernoulli trial with probability of success p_{ijt},

$$E_{\tilde{F}_{q(t)}}(M_{ijt}) = E_{\tilde{F}_{q(t)}}((1-p_{ijt})^m p_{ijt}) = [1 - E_{\tilde{F}_{q(t)}}(p_{ijt})]^m \cdot E_{\tilde{F}_{q(t)}}(p_{ijt}), \tag{23}$$

$$\begin{aligned}Var_{\tilde{F}_{q(t)}}(M_{ijt}) &= Var_{\tilde{F}_{q(t)}}((1-p_{ijt})^m p_{ijt})\\ &= \{Var_{\tilde{F}_{q(t)}}(p_{ijt}) + [E_{\tilde{F}_{q(t)}}(p_{ijt})]^2\} \times \{Var_{\tilde{F}_{q(t)}}(1-p_{ijt}) + [E_{\tilde{F}_{q(t)}}(1-p_{ijt})]^2\}^m\\ &\quad - [E_{\tilde{F}_{q(t)}}(p_{ijt})]^2 \times \{[E_{\tilde{F}_{q(t)}}(1-p_{ijt})]^2\}^m.\end{aligned} \tag{24}$$

Plugging in $\hat{E}_{\tilde{F}_{q(t)}}(p_{ijt})$ and $\widehat{Var}_{\tilde{F}_{q(t)}}(p_{ijt})$ into the above equations, one obtains $\hat{E}_{\tilde{F}_{q(t)}}(M_{ijt})$ and $\widehat{Var}_{\tilde{F}_{q(t)}}(M_{ijt})$.

alternative model 1, where $B_{ijt}(\cdot) = 0$, this reduces to $V^A_{ijt}(\Omega_t, a_{ijt} = 1; \boldsymbol{\theta}_l) = r(\boldsymbol{X}_{it}; \boldsymbol{\theta}_{Xl}) E_{\tilde{F}_t}(M_{ijt})$. For alternative model 2, $V^A_{ijt}(\Omega_t, a_{ijt} = 1; \boldsymbol{\theta}_l) = r(\boldsymbol{X}_{it}; \boldsymbol{\theta}_{Xl}) E_{\tilde{F}_t}(M_{ijt}) + \psi(\cdot) r(\boldsymbol{X}_{it}; \boldsymbol{\theta}_{Xl}) Var_{\tilde{F}_t}(M_{ijt})$. Evaluating V^A_{ijt} at $a_{ijt} = 0$, we obtain $V^A_{ijt}(\Omega_t, a_{ijt} = 0; \boldsymbol{\theta}_l) = 0$ for all models.

Moreover, with this functional form, the second constraint in equation (4) will always be guaranteed by the solution. Notice that in most models including the main model, for all $j_i < j_i' \in C_{lt}$, $V^A_{ijt}(\Omega_t, a_{ijt} = 1; \boldsymbol{\theta}_l) \geq V^A_{ij't}(\Omega_t, a_{ij't} = 1; \boldsymbol{\theta}_l)$ because the two terms only differ by $E_{\tilde{F}_t}(M_{ijt}) \geq E_{\tilde{F}_t}(M_{ij't})$. For models based on the Gittins index, the two terms also differ by $Var_{\tilde{F}_t}(M_{ijt}) \geq Var_{\tilde{F}_t}(M_{ij't})$ and the constraint continues to be satisfied. In equation (10), we add an ε_{it} to both terms and the constraint continues to be satisfied.

C.2 Specifying Choice Set C_{lt}

As discussed in Section 2, the major labs in my data received new projects through three ways with close NIH involvement: 1) a centralized planning committee periodically assigned families of novel molecules; 2) the biomedical research community nominated projects; and 3) the labs determined projects of their own interest, which they reported to the NIH well in advance. These processes placed limits on the new projects the labs could plausibly consider when they made trial allocations. These limits allow me to considerably reduce C_{lt}.

I restrict C_{lt} to include only the following project-trials. For an older project i that the lab has attempted up until trial j_i in period $t' < t$, I include trials $(j+1)_i, ..., (j+n_t)_i$ in C_{lt}. For a new project i' that the lab has not attempted until t but attempts within the next six months, I include trials $1_{i'}, ..., (n_t)_{i'}$ in C_{lt}. One can also consider using alternative windows for the new projects, such as projects attempted within the next three months or nine months. Doing so changes the magnitudes of the estimates, but all qualitative results are the same as when using six months as the window. Likewise, adding some new projects that the lab could have considered but never actually attempted could change the estimates, but the qualitative results should stay the same.

I further reduce the sizes of the choice sets used in estimation by taking random subsamples of C_{lt}. The reason is related to computation. Labs at times allocate hundreds of project-trials on a day and they usually had tens of thousands of projects in their portfolios. The sizes of some choice sets C_{lt} could be on the order of millions. Given that we have thousands of periods, if we use the full choice sets C_{lt}, we need to compute log likelihood for billions of choices in each iteration of maximum likelihood. This would result in a very large memory burden and slow computation. Moreover, it is not necessary to include every possible choice in C_{lt} to consistently estimate $\boldsymbol{\theta}_l$. A random subsample of the choices on each side of the threshold value would be sufficient. Due to the sheer number of projects in each lab's portfolio, the set of actual trials is much smaller than the set of not-allocated trials. I therefore reduce the sizes of the choice sets for estimation by taking random subsamples of the latter on the level of project and date. Let the reduced choice sets be $C^R_{lt} \subset C_{lt}$. Table C1 shows the project-trials in C^R_{lt} after random sampling from C_{lt}.

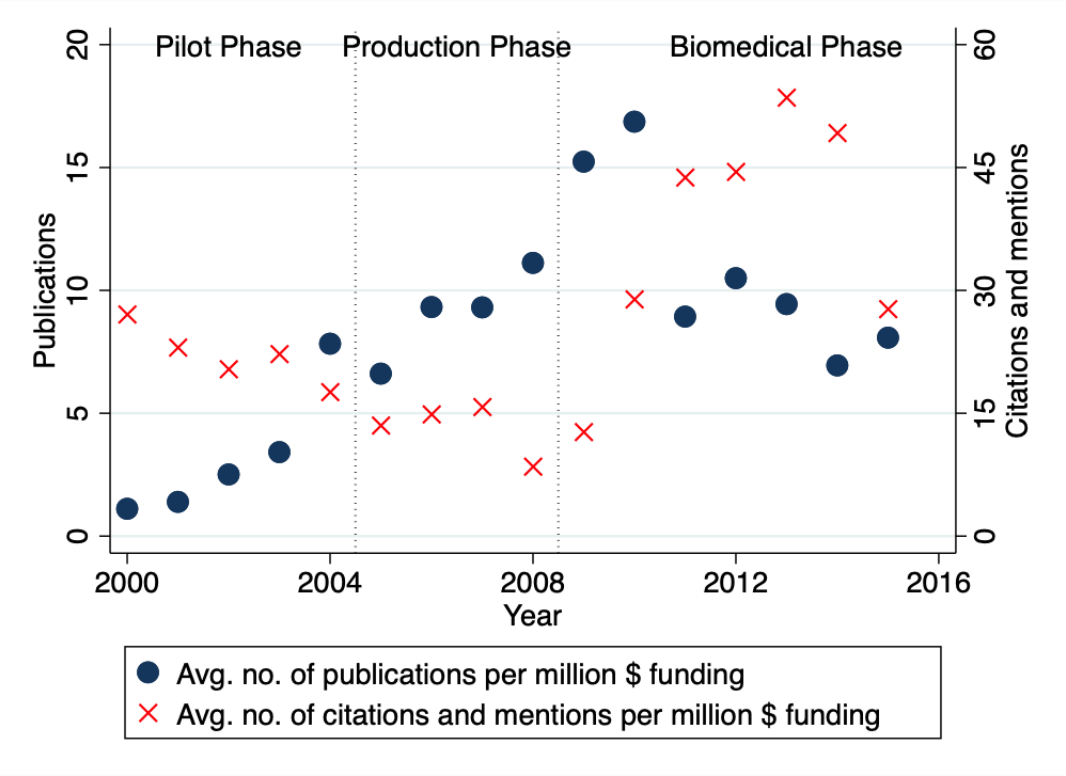
Table C1: Trials included in the reduced choice set for estimation

Project	Trial	Actually allocated on day t?	Notes
i	$(j+1)_i$	Y	Trials $(j+1)_i$ through $(j+m-1)_i$ were actually allocated on day t, include all in C^R_{lt}. When no trial was allocated on day t, m equals 1.
	⋮	⋮	
	$(j+m-1)_i$	Y	
	$(j+m)_i$	N	Trials $(j+m)_i$ through $(j+n_{lt})_i$ were not actually allocated on day t, include one random trial $(j+r)_i$ in C^R_{lt}.
	⋮	⋮	
	$(j+r)_i$	N	
	⋮	⋮	
	$(j+n_{lt})_i$	N	

Notes: For an older project that has been attempted before t, j equals the number of trials allocated to the project before t. For a new project, $j=0$. Trials in black are included in the reduced choice set C^R_{lt}. Trials in grey are in the choice set C_{lt} but are excluded from C^R_{lt} to reduce computational burden.

D Additional Results

Figure D1: Observed output: number of publications and citations



Note: The blue dots show the number of published structures on unique molecules in a given year divided by the lab consortium's funding in millions in that year. The red crosses show the number of 5-year citations and mentions the published structures in that year generated, divided by the lab consortium's funding in millions in that year. Each plot shows the average value across the four large lab in each year. The disaggregated values are in Figure 6.

Table D1: Comparison of likelihoods of different models, JCSG

Model	Free parameters in $B_{ijt}(\cdot)$	Log likelihood	Avg $\hat{P}(a^o_{ijt}=1;\boldsymbol{\theta}_l)$ actual allocation	Avg $\hat{P}(a^o_{ijt}=0;\boldsymbol{\theta}_l)$ actual nonallocation
Static	0	-1,308,021	0.691	0.891
Gittins	0	-1,104,171	0.640	0.910
UCB	1	-544,118	0.854	0.981
FlexGittins	1	-1,056,772	0.640	0.917
FlexGittins+D	2	-349,531	0.644	0.981
UCB+D	2	-206,951	0.909	0.995

Note: Estimation uses data between 2005 and 2015 from JCSG, one of the four large labs in the data. The number of trials actually allocated was 320,295. The number of trials in the choice sets after random sampling is 5,807,902. The rest of the notes of Table 2 apply.

Table D2: Comparison of likelihoods of different models, MCSG

Model	Free parameters in $B_{ijt}(\cdot)$	Log likelihood	Avg $\hat{P}(a^o_{ijt}=1;\boldsymbol{\theta}_l)$ actual allocation	Avg $\hat{P}(a^o_{ijt}=0;\boldsymbol{\theta}_l)$ actual nonallocation
Static	0	-636,089	0.644	0.994
Gittins	0	-473,481	0.669	0.996
UCB	1	-274,230	0.769	0.998
FlexGittins	1			
FlexGittins+D	2			
UCB+D	2	-157,987	0.850	0.999

Note: Estimation uses data between 2005 and 2015 from MCSG, one of the four large labs in the data. The number of trials actually allocated was 141,059. The number of trials in the choice sets after random sampling is 39,136,035. The rest of the notes of Table 2 apply.

Table D3: Comparison of likelihoods of different models, NYSGRC

Model	Free parameters in $B_{ijt}(\cdot)$	Log likelihood	Avg $\hat{P}(a^o_{ijt}=1;\boldsymbol{\theta}_l)$ actual allocation	Avg $\hat{P}(a^o_{ijt}=0;\boldsymbol{\theta}_l)$ actual nonallocation
Static	0	-677,382	0.538	0.990
Gittins	0	-550,827	0.561	0.992
UCB	1	-339,498	0.682	0.996
FlexGittins	1	-529,851	0.565	0.993
FlexGittins+D	2	-400,671	0.626	0.994
UCB+D	2	-288,162	0.700	0.996

Note: Estimation uses data between 2005 and 2015 from NYSGRC, one of the four large labs in the data. The number of trials actually allocated was 139,276. The number of trials in the choice sets after random sampling is 23,883,552. The rest of the notes of Table 2 apply.

Table D4: Out-of-sample fit of UCB+D model

Lab	Sample	Avg Log likelihood	Avg $\hat{P}(a^o_{ijt}=1;\hat{\boldsymbol{\theta}}_l)$ actual allocations	Avg $\hat{P}(a^o_{ijt}=0;\hat{\boldsymbol{\theta}}_l)$ actual nonallocations
JCSG	in	-0.033	0.909	0.994
	out	-0.036	0.914	0.996
MCSG	in	-0.004	0.826	0.999
	out	-0.004	0.871	0.999
NESG	in	-0.004	0.817	0.999
	out	-0.003	0.850	0.999
NYSGRC	in	-0.013	0.656	0.996
	out	-0.011	0.734	0.997

Note: To compute these results, I first estimate the UCB+D model using only observed allocation decisions in odd years. I then compute the in-sample results using the estimates and the odd years' decisions which I used to fit the model. I compute the out-of-sample results using even years' decisions. The rest of the notes of Table 2 apply.

Table D5: Estimates of parameters in different models, NESG

	UCB+D		Static		Gittins		UCB		FlexGittins		FlexGittins+D	
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)				
	2005–2008	2009–2015	2005–2008	2009–2015	2005–2008	2009–2015	2005–2008	2009–2015	2005–2008	2009–2015	2005–2008	2009–2015
θ_{B1}	158.3	119.5	–	–	–	–	210.5	188.8	1.6	1.4	1.4	0.8
	[156.4, 160.1]	[118.1,121.4]					[208.8, 212.5]	[188.5,189.2]	[1.6,1.6]	[1.4,1.4]	[1.4,1.4]	[0.8,0.8]
θ_{B2}	-2.3	-4.7	–	–	–	–	–	–	–	–	-6.5	-8.4
	[-2.3, -2.2]	[-4.7,-4.7]									[-6.4,-6.6]	[-8.4,-8.4]
θ_{quant}	105.6	95.9	288.9	232.6	167.5	162.3	106.1	146.1	150.0	149.5	138.4	143.5
	[105.5, 105.8]	[95.8,95.9]	[286.3, 291.4]	[232.4,232.9]	[167.1, 167.8]	[162.1,162.5]	[106.0, 106.3]	[146.1,146.2]	[149.3,151.0]	[149.4,149.8]	[137.9,139.1]	[143.3,143.7]
θ_{novel}	28.4	-23.3	202.6	20.7	67.6	-8.5	28.7	-39.8	48.4	-14.4	38.7	-7.7
	[27.8, 28.9]	[-23.4,-23.1]	[200.5, 204.3]	[20.5,20.8]	[67.3, 67.8]	[-8.6,-8.4]	[28.6, 28.9]	[-39.8,-39.8]	[47.9,49.1]	[-14.5,-14.3]	[38.1,39.4]	[-7.7,-7.7]
$\theta_{prevStructZ}$	18.1	20.1	9.8	56.5	3.8	7.9	22.3	47.6	3.0	5.2	-0.4	4.9
	[17.6, 18.6]	[20.1,20.3]	[9.4, 10.0]	[56.4,56.6]	[3.8, 3.9]	[7.8,8.0]	[22.1, 22.5]	[47.5,47.7]	[2.7,3.2]	[5.1,5.2]	[-0.7,-0.1]	[4.8,5.0]
θ_{biomed}	21.5	52.9	45.3	158.5	7.3	44.1	21.5	102.5	8.0	38.0	3.4	29.3
	[21.3, 21.7]	[52.7,52.9]	[43.1, 47.0]	[158.3,158.8]	[7.1, 7.5]	[44.0,44.2]	[21.4, 21.7]	[102.4,102.5]	[7.8,8.3]	[37.8,38.4]	[2.8,3.9]	[29.1,29.3]
$\theta_{prevPubZ}$	-9.2	-3.0	-14.5	-1.7	-6.9	-0.5	-9.4	-6.8	-6.9	-4.5	-3.8	-3.1
	[-9.3, -9.0]	[-3.1,-3.0]	[-14.7, -14.2]	[-1.7,-1.7]	[-7.0, -6.8]	[-0.5,-0.4]	[-9.6, -9.2]	[-6.8,-6.7]	[-7.2,-6.7]	[-4.5,-4.4]	[-3.9,-3.6]	[-3.1,-3.1]
θ_{human}	67.1	124.0	105.9	191.0	51.4	90.5	69.8	160.6	40.2	83.0	36.3	74.5
	[66.9, 67.3]	[123.9,124.0]	[104.8, 106.9]	[190.7,191.3]	[51.2, 51.6]	[90.4,90.6]	[69.7, 70.0]	[160.6,160.7]	[40.1,40.5]	[82.9,83.1]	[36.0,36.7]	[74.5,74.6]
$\theta_{eukaryote}$	-29.8	-35.9	67.7	61.6	-8.3	-26.9	-30.0	-0.3	-14.3	-33.7	-11.7	-33.4
	[-29.9, -29.7]	[-36.0,-35.8]	[67.1, 69.1]	[61.5,61.8]	[-8.7, -8.0]	[-27.0,-26.8]	[-30.4, -29.8]	[-0.3,-0.2]	[-14.4,-13.9]	[-33.7,-33.6]	[-12.0,-11.6]	[-33.4,-33.4]
$\theta_{membrane}$	58.3	13.1	96.5	52.3	35.2	28.4	58.1	22.7	31.2	24.8	23.5	22.1
	[58.0, 58.6]	[13.1,13.3]	[96.1, 97.2]	[52.1,52.4]	[35.1, 35.4]	[28.3,28.5]	[57.9, 58.5]	[22.7,22.8]	[31.1,31.4]	[24.8,24.9]	[23.1,23.9]	[22.1,22.2]

Note: Table displays the full estimates of parameters in different models for NESG, one of the four large labs. Results from other labs are available upon request. $prevStructZ_{iy}$ represents the number of standard deviations by which $prevStruct_{iy}$ differs from the yearly mean $\overline{prevStruct}_y$. $prevPubZ_{iy}$ represents the number of standard deviations by which $prevPub_{iy}$ differs from the yearly mean $\overline{prevPub}_y$. The rest of the notes of Table 3 apply.

Table D6: Estimates of main parameters of interest in UCB+D model, other labs

	JCSG		MCSG		NYSGRC	
	2005–2008	2009–2015	2005–2008	2009–2015	2005–2008	2009–2015
Parameter	(1)	(2)	(3)	(4)	(5)	(6)
θ_{B1}	558.2	1001.9	273.3	127.1	61.6	115.6
	[551.1,573.4]	[977.6,1028.0]	[266.2,284.8]	[126.3,127.8]	[61.2,61.9]	[114.2,116.9]
θ_{B2}	-247.9	-104.7	-3.9	-3.8	-2.9	-3.9
	[-247.6, -248.5]	[-102.9, -106.6]	[-3.9,-3.9]	[-3.8,-3.8]	[-2.9,-3.0]	[-3.9,-3.9]
θ_{biomed}	-34.3	105.1	12.7	84.8	33.0	89.4
	[-34.6,-34.1]	[104.7,105.8]	[12.6,12.7]	[84.8,84.8]	[32.9,33.1]	[89.1,89.8]

Note: Table displays the estimates of the main parameters of interest in the UCB+D model for the other three large labs. Full estimates of parameters in different models are available upon request. 95% confidence intervals are computed using the MCMC approach in Chernozhukov & Hong (2003) and are shown in brackets. These confidence intervals are almost identical to those computed using Procedure 1 of Chen et al. (2018).

Table D7: Simulated outcomes of UCB+D model, other labs

Lab	Model	Projects attempted	Unique publications	Citations	Downloads (millions)
JCSG	UCB+D	40,881	1,607	1,495	38.4
	Actual	40,881	1,512	1,463	36.3†
MCSG	UCB+D	77,503	2,040	2,524	46.3
	Actual	78,740	2,276	3,145	50.0†
NYSGRC*	UCB+D	59,734	626	2,579	14.7
	Actual	59,734	617	2,575	14.4†

Note: Each simulation uses $\hat{\boldsymbol{\theta}}_l$ from parameter estimates of the corresponding model. Results are averaged from three simulations of each model. †The download data on actually published projects are between 2007 and 2013, so actual five-year downloads may not be available for some projects. I predict five-year downloads for the actually published projects using the predictive model described in Appendix B.3. *I note data problems related to NYSGRC. For this particular lab, more than half of the trials that produced structures either miss key stage dates or have those dates in wrong orders (for example, publication is at an earlier date than previous stages). As a result, I am not able to correctly simulate the dates and outcomes of different stages of trials. Though this set of simulation results looks quite nice, please take them with a grain of salt.

Table D8: Out-of-sample simulation results for UCB+D model

Lab	Model	Projects attempted	Unique publications	Citations	Downloads (millions)
JCSG	in	40,881	1,607	1,495	38.4
	out	40,881	1,621	1,481	38.6
	actual	40,881	1,512	1,463	36.3†
MCSG	in	77,503	2,040	2,524	46.3
	out	77,504	2,051	2,553	46.4
	actual	78,740	2,276	3,145	50.0†
NESG	in	59,947	1,097	3,376	25.6
	out	59,913	1,085	3,336	25.3
	actual	59,953	1,053	3,502	24.5†
NYSGRC*	in	59,734	626	2,579	14.7
	out	59,734	628	2,594	14.7
	actual	59,734	617	2,575	14.4†

Note: In-sample simulation results are identical to those for the UCB+D model in Tables 4 and D7. I simulate the out-of-sample results as follows. I first fit the UCB+D model for each lab with odd years of observed decisions. Using those estimated parameters, I simulate each lab's full input allocation history and output, in odd and even years. The rest of the notes of Table D7 apply.

Table D9: Counterfactual outcomes, no exploration, other labs

Lab	Counterfactual model	Projects attempted	Unique publications	Citations	Downloads (millions)
JCSG	Static	10,710	1,338	1,270	32.4
		(-74%)	(−17%)	(−15%)	(−16%)
	Baseline model	40,881	1,607	1,495	38.4
MCSG	Static	14,883	668	856	15.6
		(-81%)	(−67%)	(−66%)	(−66%)
	Baseline model	77,503	2,040	2,524	46.3
NYSGRC*	Static	5,203	247	1,026	5.9
		(-91%)	(−61%)	(−60%)	(−60%)
	Baseline model	59,734	626	2,579	14.7

Note: Each simulation uses $\hat{\boldsymbol{\theta}}_{Xl}$ from parameter estimates of the UCB+D model for the corresponding lab. These estimates are available upon request. Results are averaged from three simulations of the model. Output from the baseline model are identical to those for the UCB+D model in Appendix Table D7. Parentheses show percentage differences as compared to the baseline model. Table shows results from the three other large labs. See Table 5 for results from NESG. *I note data problems related to NYSGRC. For this particular lab, more than half of the trials that produced structures either miss key stage dates or have those dates in wrong orders (for example, publication is at an earlier date than previous stages). As a result, I am not able to correctly simulate the dates and outcomes of different stages of trials. Though this set of simulation results looks quite nice, please take them with a grain of salt.