Using an existing metabolic model to (start to) build a new one!

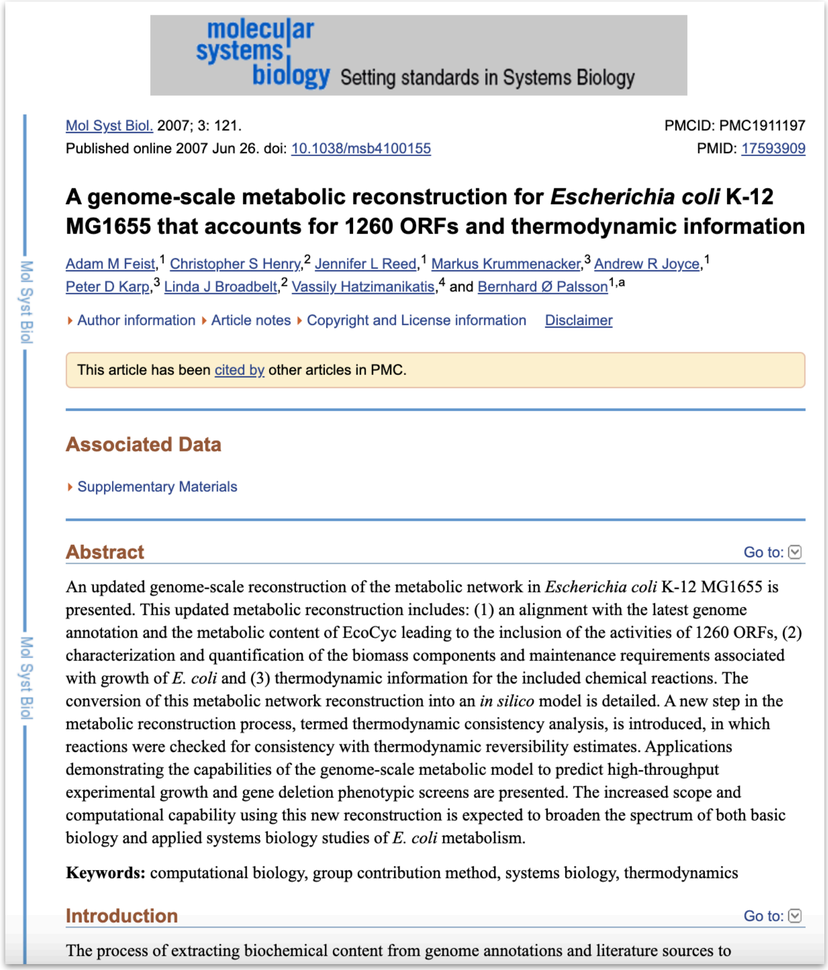
Dr Sandy Macdonald, Bioscience Technology Facility Bioinformatician,
Friday, 19th November, 2021

Where to start?

- Let's imagine you have an annotated genome sequence for a bacterium and want to build a flux balance analysis (FBA) model for it. Where do you start?
- You could go through the genome, gene by gene, search for that gene in KEGG or MetaCyc to find what reaction it catalyses, and then add the reactions, metabolites, and genes to your model, typing out the SBML by hand. 🤪 *(I did this in my postdoc in 2008)*
- Starting from an existing model/s and transferring as much knowledge from that model as possible to your new model would be much better!
- The result is not a complete model, but a very good starting point

E. coli as a starting point

- As with many other things, *E. coli* is a good starting point, because:
  - Its genome is very well annotated
  - Its metabolism is (relatively) well understood
  - Many of its metabolic enzymes have been studied experimentally
  - There have been several versions now of *E. coli* FBA models, each time improved and with more genes included



molecular systems biology Setting standards in Systems Biology

Mol Syst Biol. 2007; 3: 121.
Published online 2007 Jun 26. doi: 10.1038/msb4100155
PMCID: PMC1911197
PMID: 17593909

A genome-scale metabolic reconstruction for *Escherichia coli* K-12 MG1655 that accounts for 1260 ORFs and thermodynamic information

Adam M Feist,[1] Christopher S Henry,[2] Jennifer L Reed,[1] Markus Krummenacker,[3] Andrew R Joyce,[1] Peter D Karp,[3] Linda J Broadbelt,[2] Vassily Hatzimanikatis,[4] and Bernhard Ø Palsson[1,a]

Associated Data

Supplementary Materials

Abstract Go to:

An updated genome-scale reconstruction of the metabolic network in *Escherichia coli* K-12 MG1655 is presented. This updated metabolic reconstruction includes: (1) an alignment with the latest genome annotation and the metabolic content of EcoCyc leading to the inclusion of the activities of 1260 ORFs, (2) characterization and quantification of the biomass components and maintenance requirements associated with growth of *E. coli* and (3) thermodynamic information for the included chemical reactions. The conversion of this metabolic network reconstruction into an *in silico* model is detailed. A new step in the metabolic reconstruction process, termed thermodynamic consistency analysis, is introduced, in which reactions were checked for consistency with thermodynamic reversibility estimates. Applications demonstrating the capabilities of the genome-scale metabolic model to predict high-throughput experimental growth and gene deletion phenotypic screens are presented. The increased scope and computational capability using this new reconstruction is expected to broaden the spectrum of both basic biology and applied systems biology studies of *E. coli* metabolism.

Keywords: computational biology, group contribution method, systems biology, thermodynamics

Introduction Go to:

The process of extracting biochemical content from genome annotations and literature sources to

https://pubmed.ncbi.nlm.nih.gov/17593909/

Assigning gene function

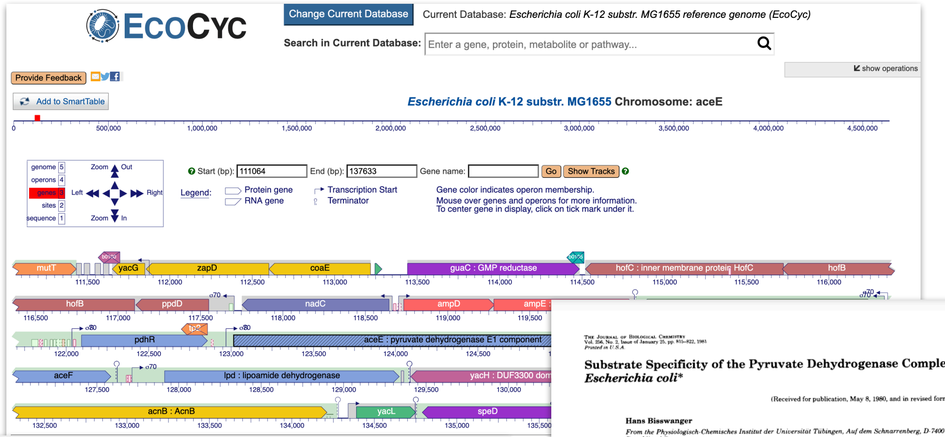
- How do we know the function of a gene?

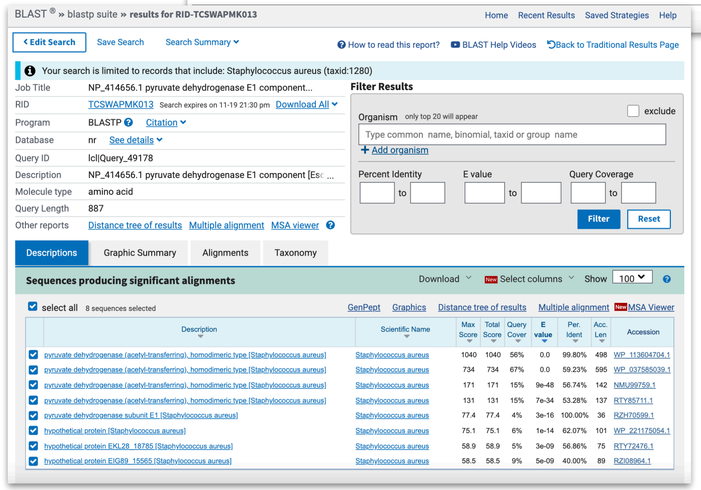
1. Genome sequencing, finding open reading frames (start/stop codons), and hence coding sequences for genes

2. Assign gene functions based on sequence similarity (homology) to other annotated genome sequences

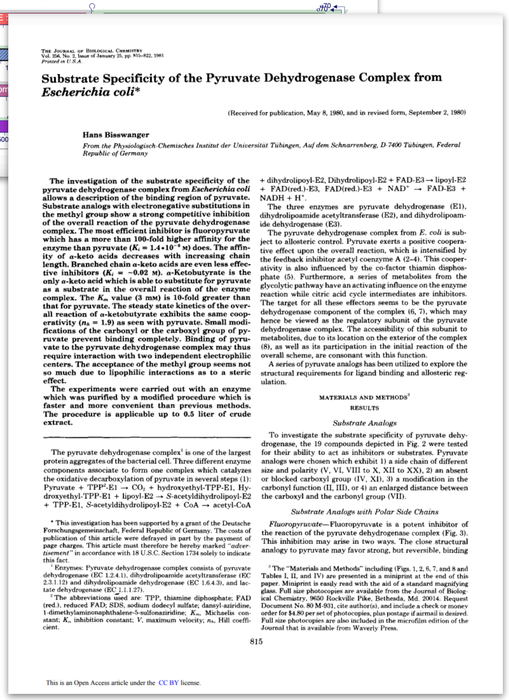
3. Expressing protein, characterising it experimentally, determining protein structure

https://biocyc.org/gene?orgid=ECOLI&id=EG10024





https://blast.ncbi.nlm.nih.gov/Blast.cgi



THE JOURNAL OF BIOLOGICAL CHEMISTRY
Vol. 256, No. 1, Issue of January 25, pp. 815–822, 1981
Printed in U.S.A.

Substrate Specificity of the Pyruvate Dehydrogenase Complex from *Escherichia coli**

(Received for publication, May 8, 1980, and in revised form, September 2, 1980)

Hans Bisswanger
From the Physiologisch-Chemisches Institut der Universität Tübingen, Auf dem Schnarrenberg, D-7400 Tübingen, Federal Republic of Germany

The investigation of the substrate specificity of the pyruvate dehydrogenase complex from *Escherichia coli* allows a description of the binding region of pyruvate. Substrate analogs with electronegative substitutions in the methyl group show a strong competitive inhibition of the overall reaction of the pyruvate dehydrogenase complex. The most efficient inhibitor is fluoropyruvate which has a more than 100-fold higher affinity for the enzyme than pyruvate (K_i = 1.4 × 10⁻⁴ M) does. The affinity of α-keto acids decreases with increasing chain length. Branched chain α-keto acids are even less effective inhibitors (K_i = ~0.02 M). α-Ketobutyrate is the only α-keto acid which is able to substitute for pyruvate as a substrate in the overall reaction of the enzyme complex. The K_m value (3 mM) is 10-fold greater than that for pyruvate. The steady state kinetics of the overall reaction of α-ketobutyrate exhibits the same cooperativity (n_H = 1.9) as seen with pyruvate. Small modifications of the carbonyl or the carboxyl group of pyruvate prevent binding completely. Binding of pyruvate to the pyruvate dehydrogenase complex may thus require interaction with two independent electrophilic centers. The acceptance of the methyl group seems not so much due to lipophilic interactions as to a steric effect.

The experiments were carried out with an enzyme which was purified by a modified procedure which is faster and more convenient than previous methods. The procedure is applicable up to 0.5 liter of crude extract.

The pyruvate dehydrogenase complex[1] is one of the largest protein aggregates of the bacterial cell. Three different enzyme components associate to form one complex which catalyzes the oxidative decarboxylation of pyruvate in several steps (1): Pyruvate + TPP^{E1}-E1 → CO_2 + hydroxyethyl-TPP-E1, Hydroxyethyl-TPP-E1 + lipoyl-E2 → S-acetyldihydrolipoyl-E2 + TPP-E1, S-acetyldihydrolipoyl-E2 + CoA → acetyl-CoA + dihydrolipoyl-E2, Dihydrolipoyl-E2 + FAD-E3 → lipoyl-E2 + FADred.)-E3, FADred.)-E3 + NAD^+ → FAD-E3 + $NADH + H^+$.

The three enzymes are pyruvate dehydrogenase (E1), dihydrolipoamide acetyltransferase (E2), and dihydrolipoamide dehydrogenase (E3).

The pyruvate dehydrogenase complex from *E. coli* is subject to allosteric control. Pyruvate exerts a positive cooperative effect upon the overall reaction, which is intensified by the feedback inhibitor acetyl coenzyme A (2–4). This cooperativity is also influenced by the co-factor thiamin diphosphate (5). Furthermore, a series of metabolites from the glycolytic pathway have an activating influence on the enzyme reaction while citric acid cycle intermediates are inhibitors. The target for all these effectors seems to be the pyruvate dehydrogenase component of the complex (6, 7), which may hence be viewed as the regulatory subunit of the pyruvate dehydrogenase complex. The accessibility of this subunit to metabolites, due to its location on the exterior of the complex (8), as well as its participation in the initial reaction of the overall scheme, are consonant with this function.

A series of pyruvate analogs has been utilized to explore the structural requirements for ligand binding and allosteric regulation.

MATERIALS AND METHODS[2]

RESULTS

Substrate Analogs

To investigate the substrate specificity of pyruvate dehydrogenase, the 19 compounds depicted in Fig. 2 were tested for their ability to act as inhibitors or substrates. Pyruvate analogs were chosen which exhibit 1) a side chain of different size and polarity (V, VI, VIII to X, XII to XX), 2) an absent or blocked carboxyl group (IV, XI), 3) a modification in the carbonyl function (II, III), or 4) an enlarged distance between the carboxyl and the carbonyl group (VII).

Substrate Analogs with Polar Side Chains

Fluoropyruvate—Fluoropyruvate is a potent inhibitor of the reaction of the pyruvate dehydrogenase complex (Fig. 3). This inhibition may arise in two ways. The close structural analogy to pyruvate may favor strong, but reversible, binding

* This investigation has been supported by a grant of the Deutsche Forschungsgemeinschaft, Federal Republic of Germany. The costs of publication of this article were defrayed in part by the payment of page charges. This article must therefore be hereby marked "*advertisement*" in accordance with 18 U.S.C. Section 1734 solely to indicate this fact.

[1] Enzymes: Pyruvate dehydrogenase complex consists of pyruvate dehydrogenase (EC 1.2.4.1), dihydrolipoamide acetyltransferase (EC 2.3.1.12) and dihydrolipoamide dehydrogenase (EC 1.6.4.3), and lactate dehydrogenase (EC 1.1.1.27).

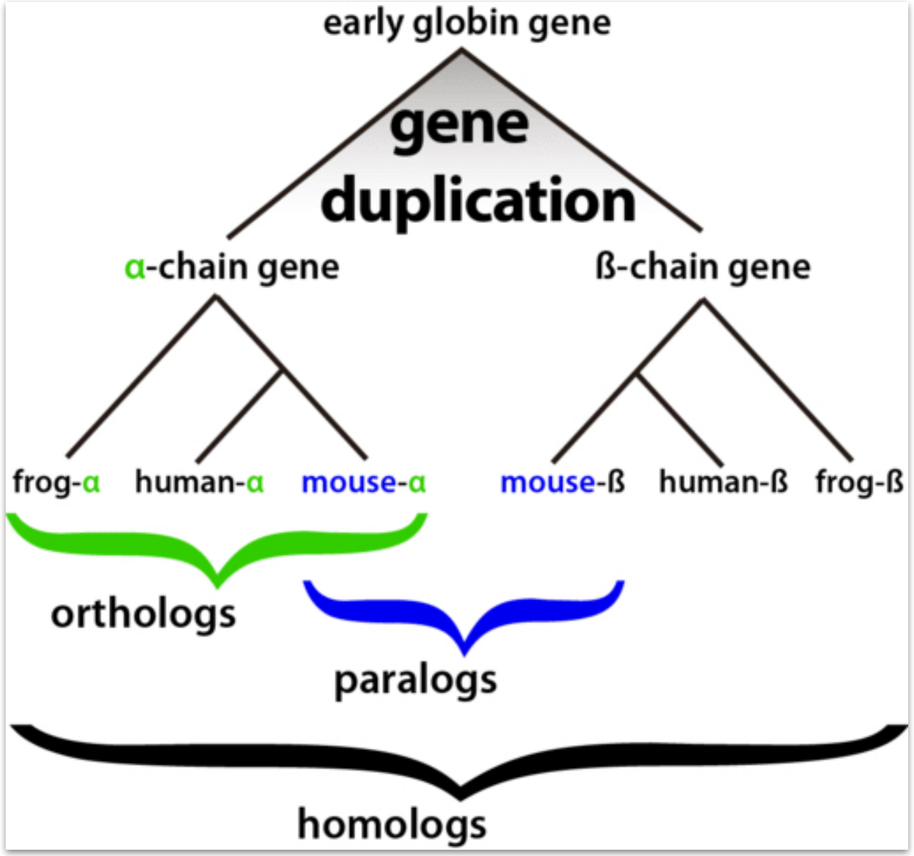
[2] The abbreviations used are: TPP, thiamine diphosphate; FAD (red.), reduced FAD; SDS, sodium dodecyl sulfate; dansyl-aziridine, 1-dimethylaminonaphthalene-5-sulfonaziridine; K_m, Michaelis constant; K_i, inhibition constant; V, maximum velocity; n_H, Hill coefficient.

https://www.jbc.org/article/S0021-9258(19)70050-7/pdf

Homologs, orthologs, and paralogs

What's the difference?

- "Paralogs" arise from a gene duplication event in an organism or a distant ancestor and often acquire a different function
- "Orthologs" occur from a speciation event, so orthologs in two species, e.g. human and mouse α-globins, will share a function
- "Homologs" can describe both paralogs and orthologs, but it's a vague term and I wouldn't encourage its use!
- **We're looking for orthologs**
- There are also analogs, xenologs, and more!
- Find out more here: https://en.wikipedia.org/wiki/Sequence_homology



https://bitesizebio.com/26762/homology-terminology-never-say-wrong-word/

What are the pitfalls?

- Is your genome similar enough to *E. coli* to transfer gene functions?
  - **Use a more closely-related, well-characterised genome**
- Could two proteins share similar sequence but have different functions?
  - **Yes! These could be paralogs or promiscuous enzymes**
- What if we match against a protein that shares a domain (part of the sequence) but has a different overall function?
  - **We can set thresholds on our sequence searches to reduce the likelihood of this: % identity and % coverage**
- Finding the true ortholog is not easy!
  - **Building a phylogenetic tree is the best way to work this out**

Picking the right starting point

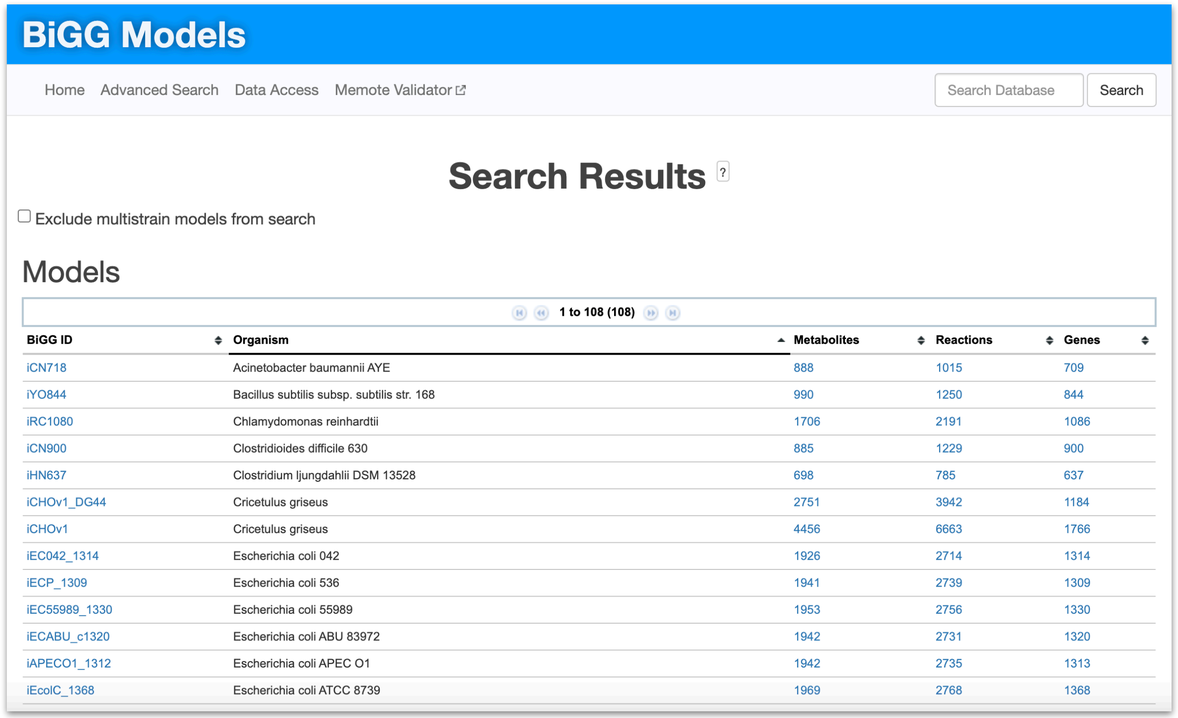
So, we need to do the following:

1. Pick a closely related (to our "unknown" organism), well-characterised genome, with an FBA model, as our starting point.
2. Carefully match gene/protein sequences from our "known" organism against our "unknown" organism's gene/protein sequences (using BLAST).
3. Filter down the set of reactions in the "known" FBA model to only those with matches/hits in our "unknown" organism.

And that's just the start!

The BiGG database

- *Bio*chemical, *G*enetic, and *G*enomic database
- 108 FBA models, 48 single strain models, 30 species
- Includes human, mouse, yeast, parasites, quite a few species of bacteria
- 58 *E. coli* models!! (more than half of the models in the database)
- There are lots more FBA models elsewhere (e.g. with publications), but...
- ...the models in BiGG follow sensible rules for naming reactions and metabolites
- They have an API for grabbing data!

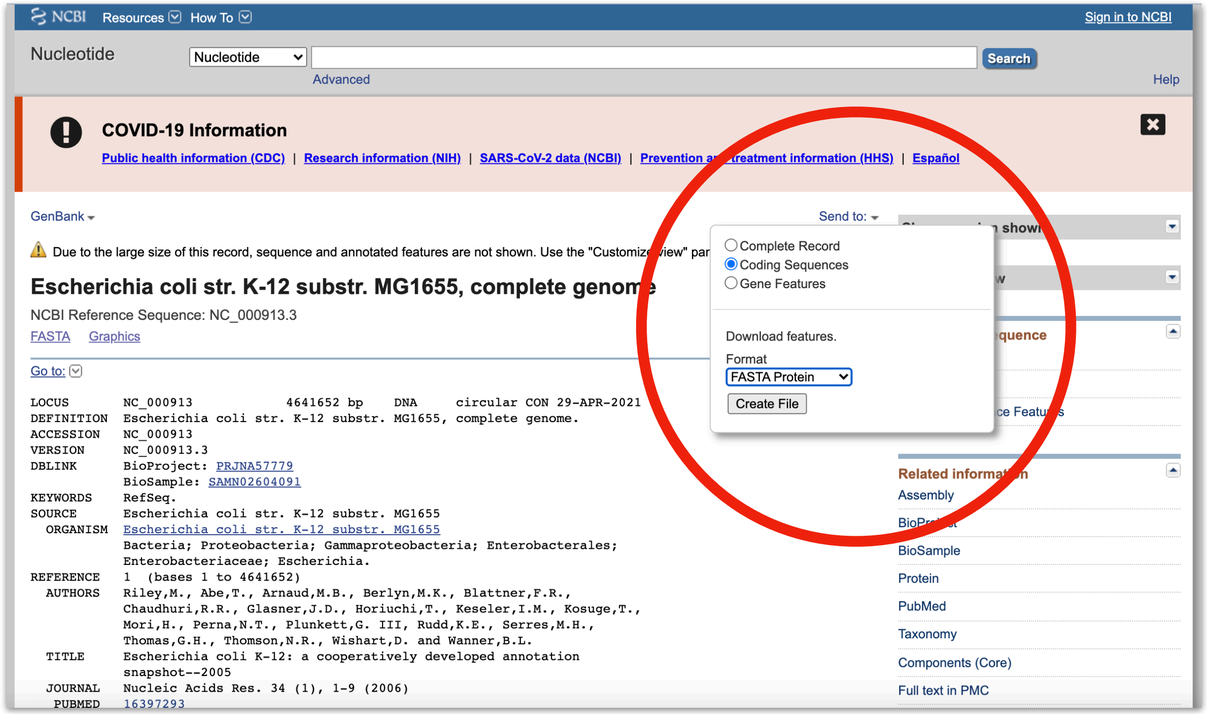


BiGG ID	Organism	Metabolites	Reactions	Genes
iCN718	Acinetobacter baumannii AYE	888	1015	709
iYO844	Bacillus subtilis subsp. subtilis str. 168	990	1250	844
iRC1080	Chlamydomonas reinhardtii	1706	2191	1086
iCN900	Clostridioides difficile 630	885	1229	900
iHN637	Clostridium ljungdahlii DSM 13528	698	785	637
iCHOv1_DG44	Cricetulus griseus	2751	3942	1184
iCHOv1	Cricetulus griseus	4456	6663	1766
iEC042_1314	Escherichia coli 042	1926	2714	1314
iECP_1309	Escherichia coli 536	1941	2739	1309
iEC55989_1330	Escherichia coli 55989	1953	2756	1330
iECABU_c1320	Escherichia coli ABU 83972	1942	2731	1320
iAPECO1_1312	Escherichia coli APEC O1	1942	2735	1313
iEcolC_1368	Escherichia coli ATCC 8739	1969	2768	1368

http://bigg.ucsd.edu/models

Finding the "known" protein sequences

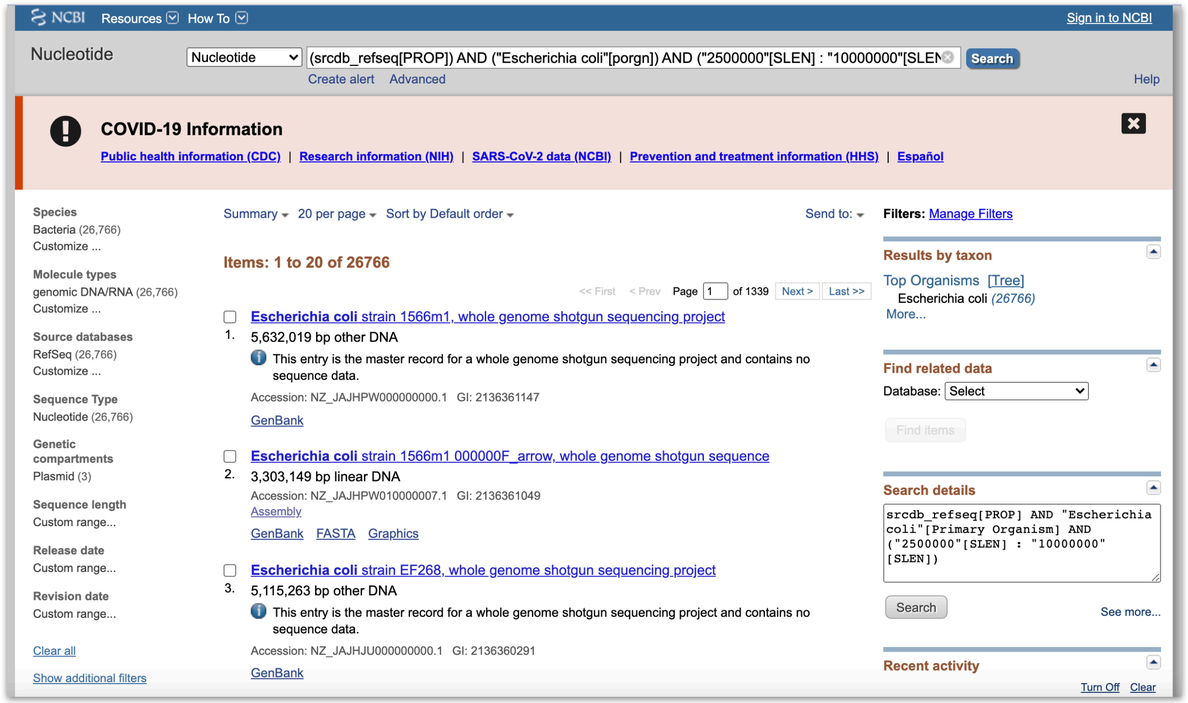
- The BiGG model page should have a link to the genome sequence for the "known" species on NCBI
- Download the **coding sequences** in **FASTA Protein** format, making sure to give the file a sensible name (the default is "sequence")
- Some of you will be using the *E. coli* K-12 MG1655 FBA model (called **iAF1260**), and you should use the link on the right to get the protein sequences
- The others are using the *S. aureus* N315 FBA model (called **iSB619**), and you should use this link to get your protein sequences: https://www.ncbi.nlm.nih.gov/nuccore/NC_002745.2



https://www.ncbi.nlm.nih.gov/nuccore/NC_000913.3

Finding the "unknown" protein sequences

- You'll need to pick a closely related species or another strain of the same species as your "known" genome/ model
- The best place to find these in the NCBI nucleotide database using the filtering options or advanced search to limit it to the right species and RefSeq genomes that are "high quality".
- https://www.ncbi.nlm.nih.gov/nuccore
- Make sure to ignore plasmids. You can do this with the advanced search by giving a sequence length that is in a similar range to your "known" genome, e.g. 2,500,000 - 10,000,000 for *E. coli*
- Also watch out for the records that say "This entry is the master record... and contains no sequence data."
- I used the advanced search: `(srcdb_refseq[PROP]) AND ("Escherichia coli"[porgn]) AND ("2500000"[SLEN] : "10000000"[SLEN])`



https://www.ncbi.nlm.nih.gov/nuccore?term=%28srcdb_refseq%5BPROP%5D%29%20AND%20%28%22Escherichia%20coli%22%5Bporgn%5D%29%20AND%20%28%222500000%22%5BSLEN%5D%20%3A%20%2210000000%22%5BSLEN%5D%29&cmd=DetailsSearch

Reciprocal BLAST hits (RBH)

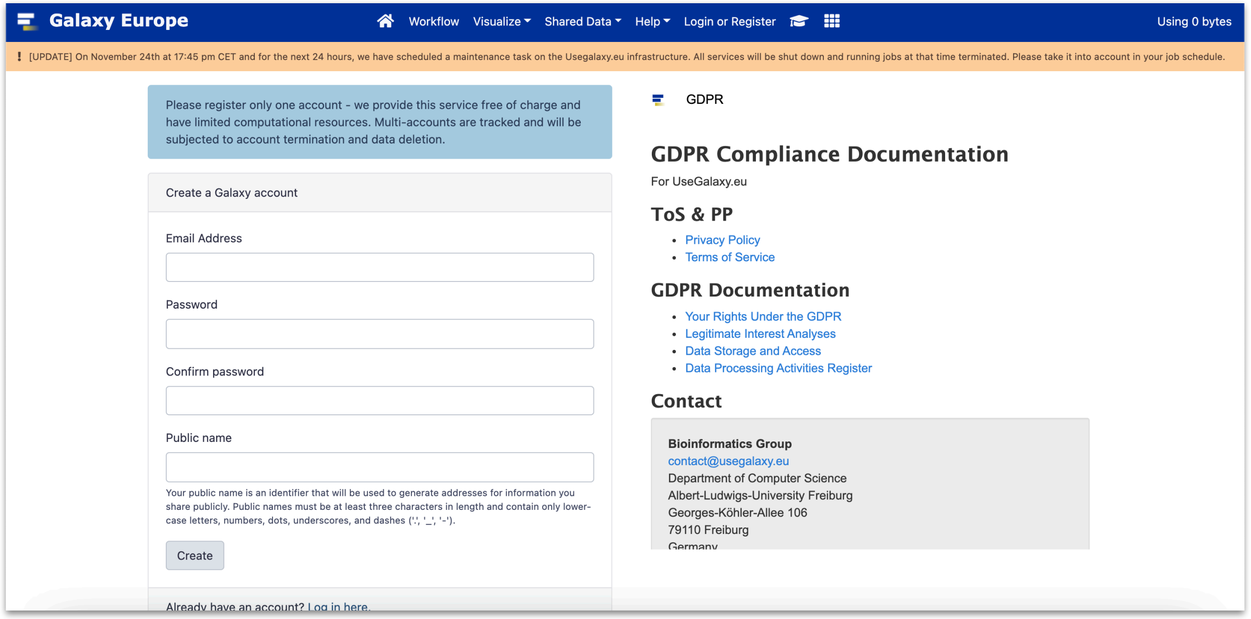
- **BLAST** is a sequence similarity search method that's widely used
- Usually you'll search one or more sequences against a database of sequences to find the most similar matches (hits)
- You measure similarity by:
  - **% identity** (prop. of matching letters in the alignment)
  - **% coverage** (prop. of your query sequence covered by the alignment)
  - E value (probability that the match in the database occurred at random)
- A reciprocal BLAST hit BLASTs the hit back against all of the original sequences to ensure the original sequence is found again

Galaxy (usegalaxy.eu)

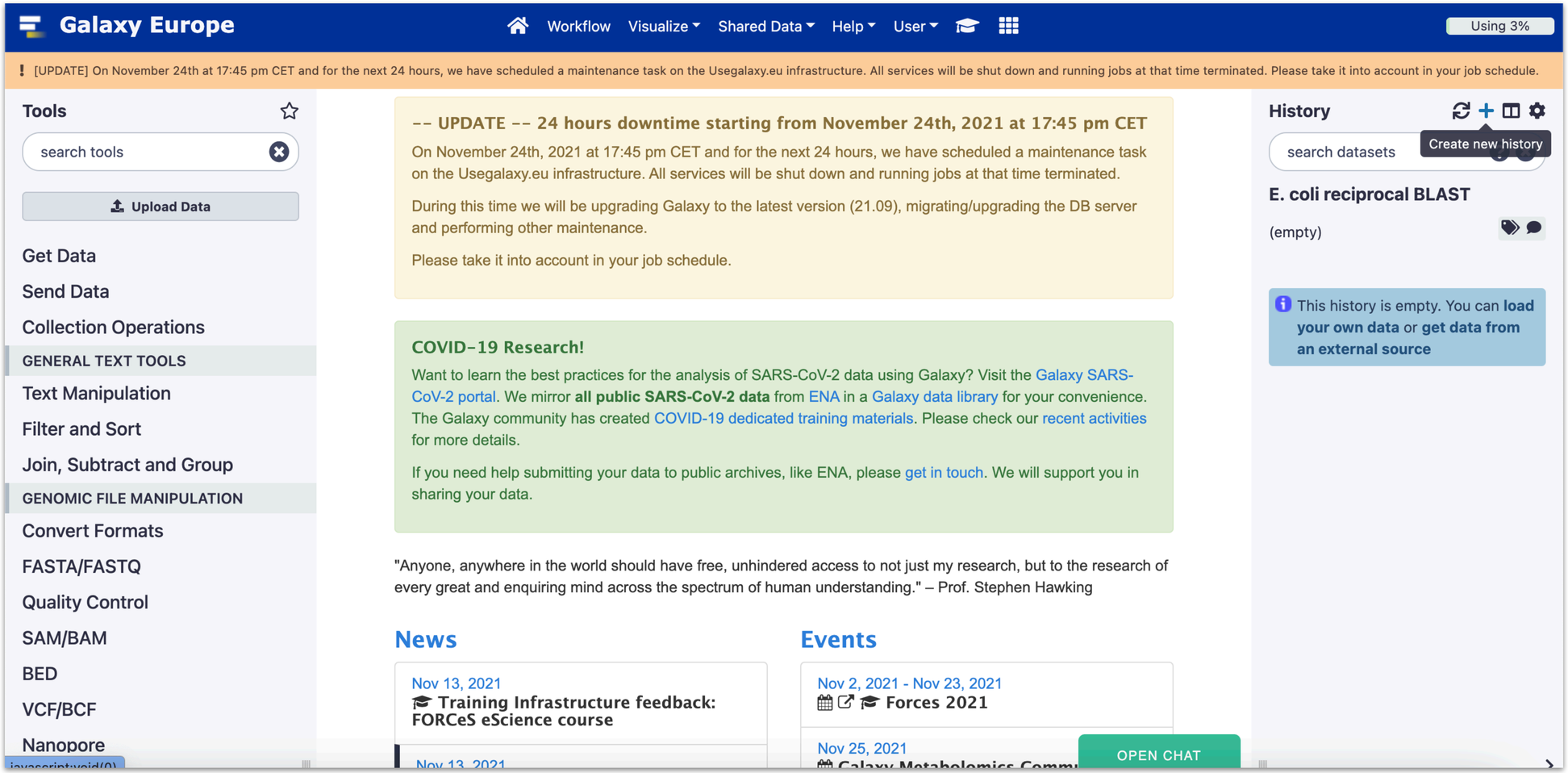
- Galaxy is a web tool for running bits of bioinformatics software, like BLAST, that would normally be on the command line

- You can sign up for a free account here: https://usegalaxy.eu/login

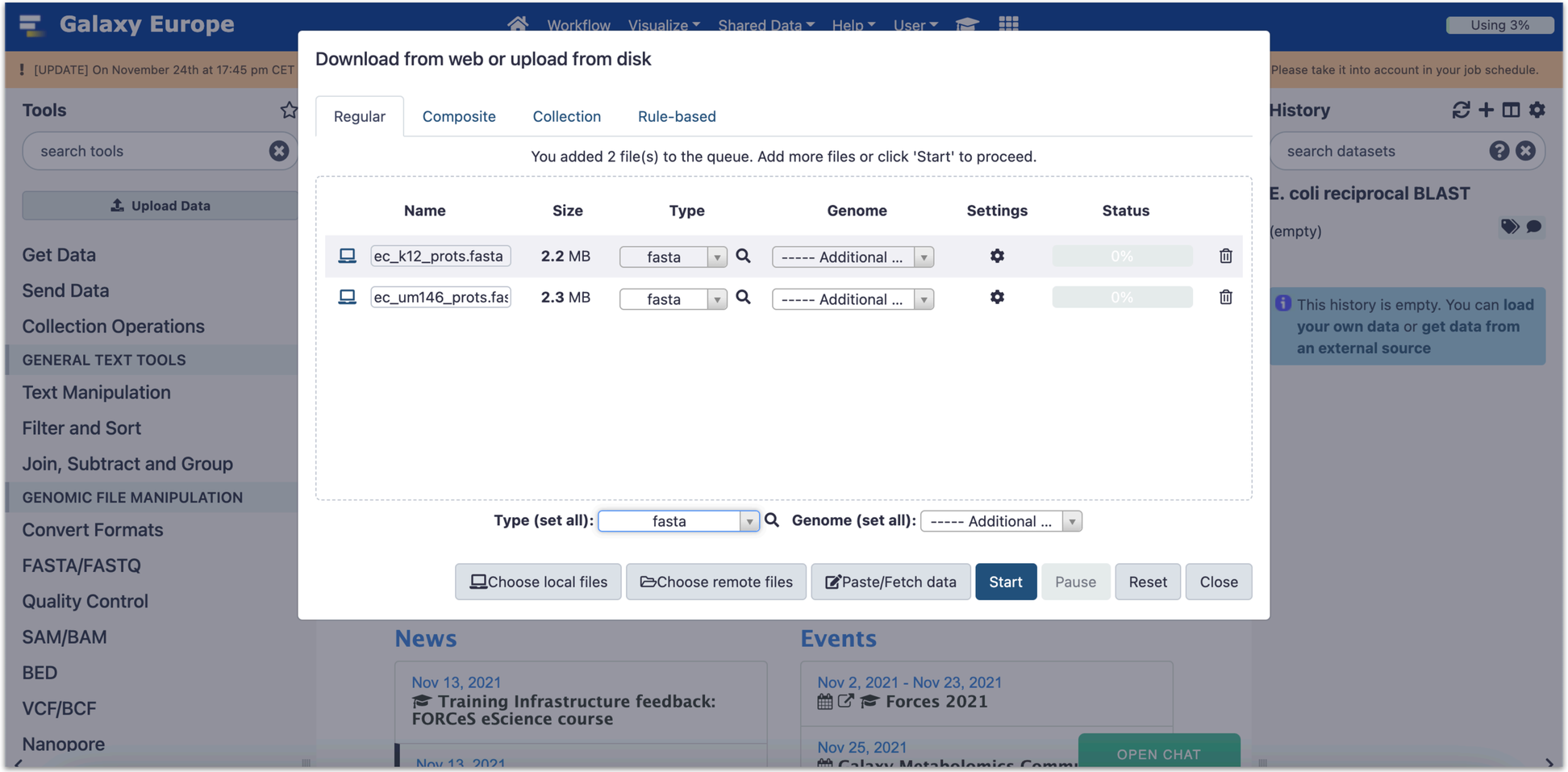
- **Galaxy has a reciprocal BLAST function that we can use with our two sets of protein sequences to find matches**



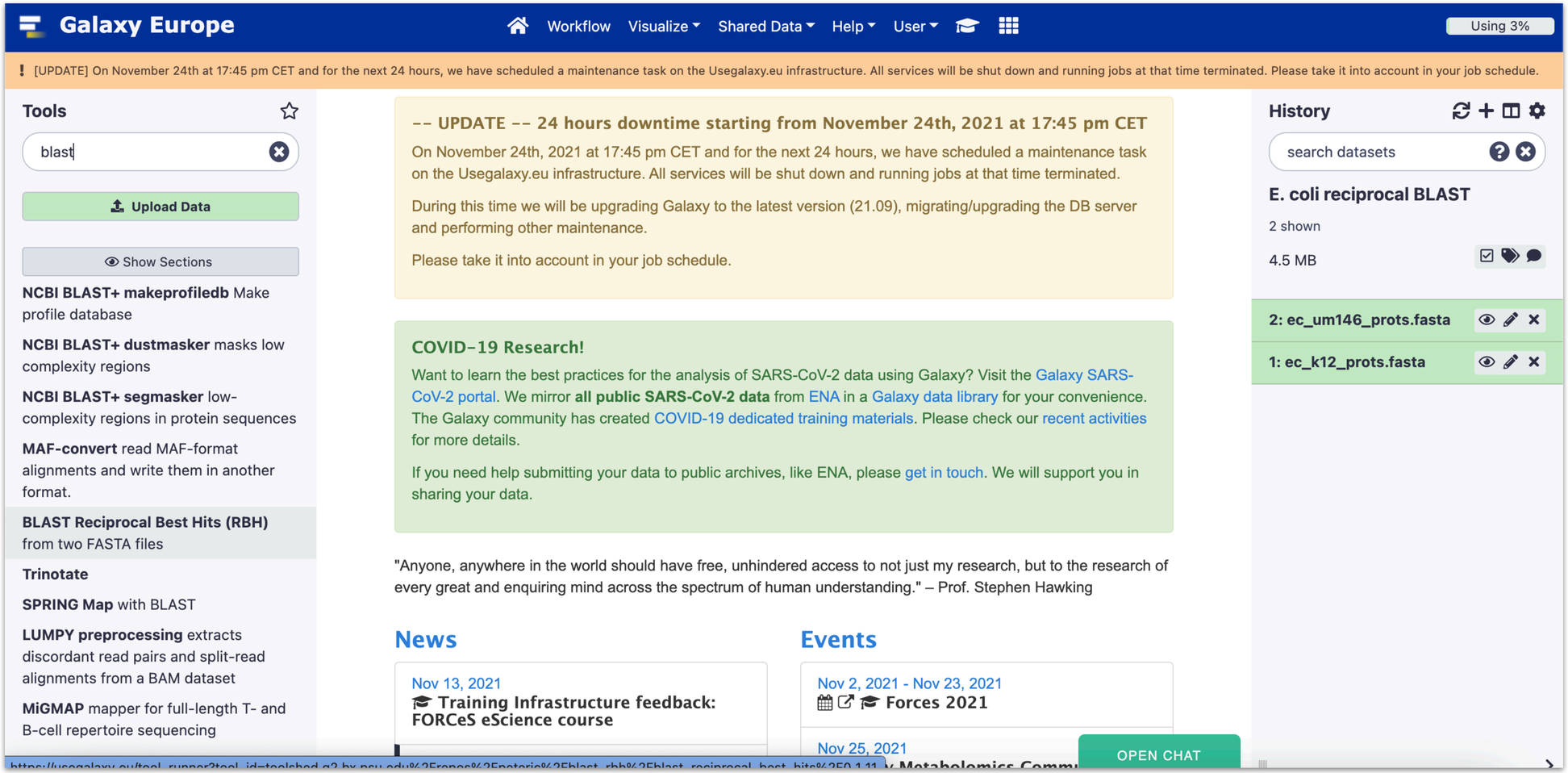
Create a new history



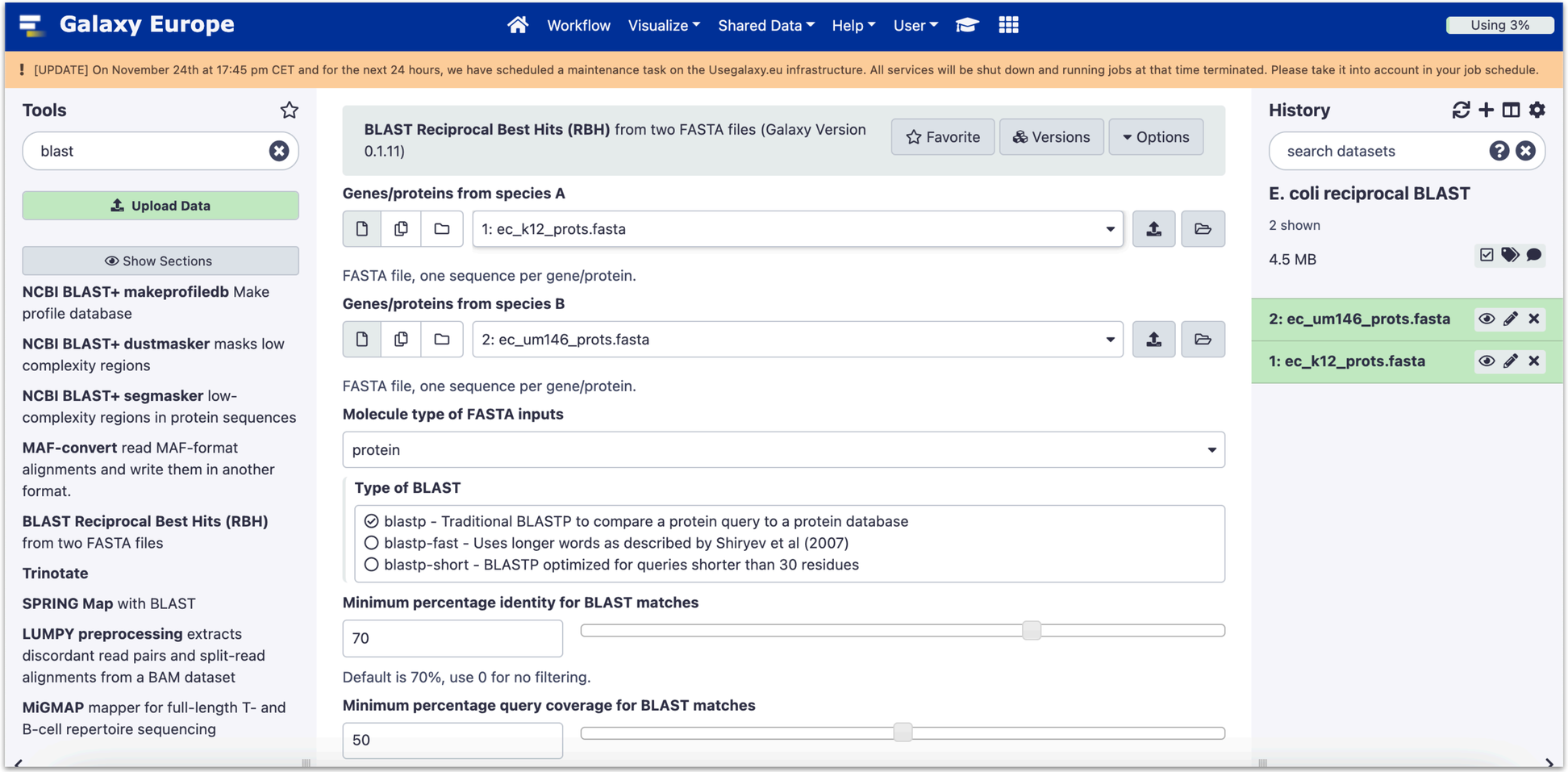
Click on "load your own data", select your two fasta files, and set type to fasta, click "Start"



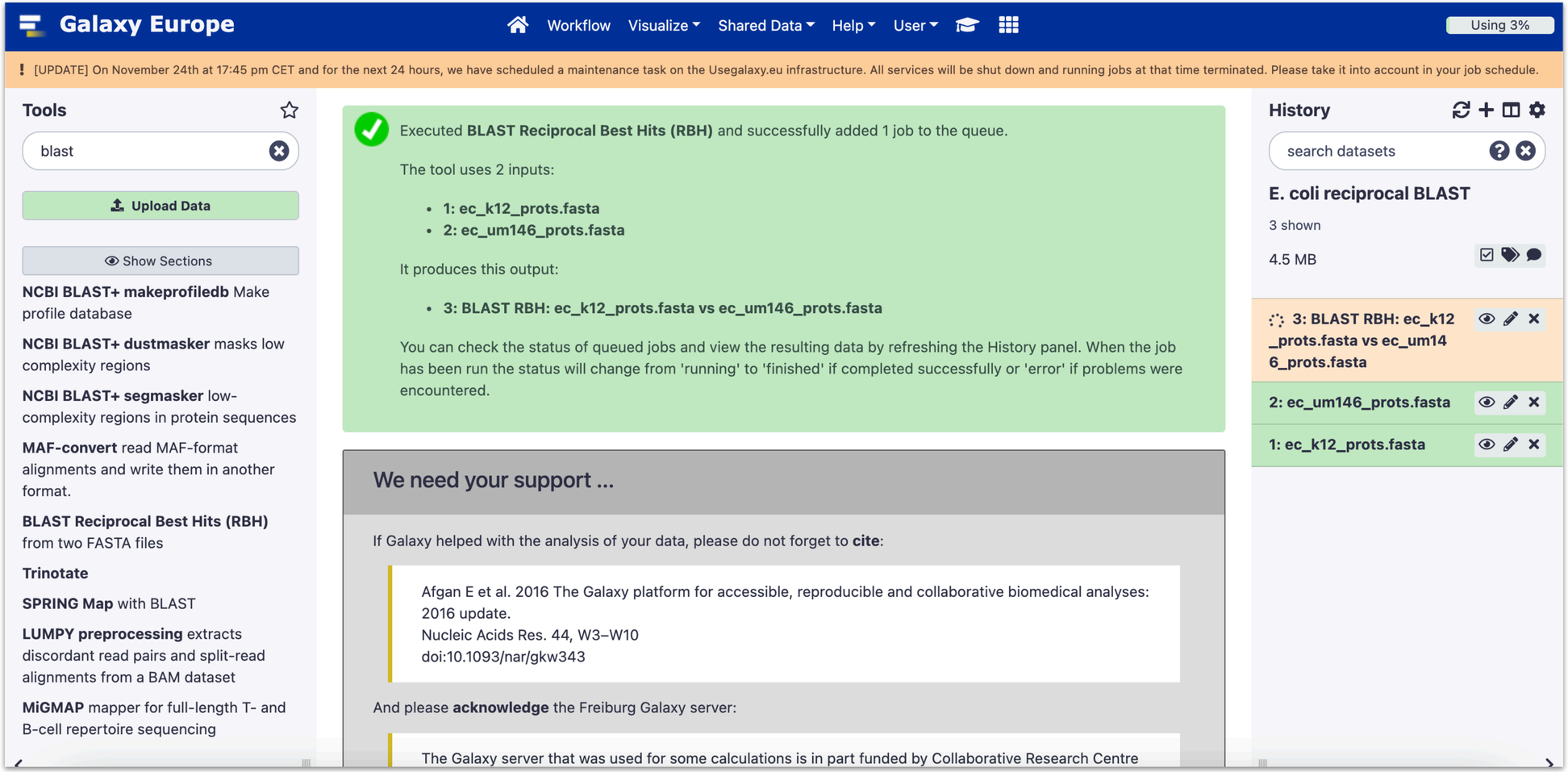
Search the Tools for "blast" and select "BLAST Reciprocal Best Hits (RBH)"



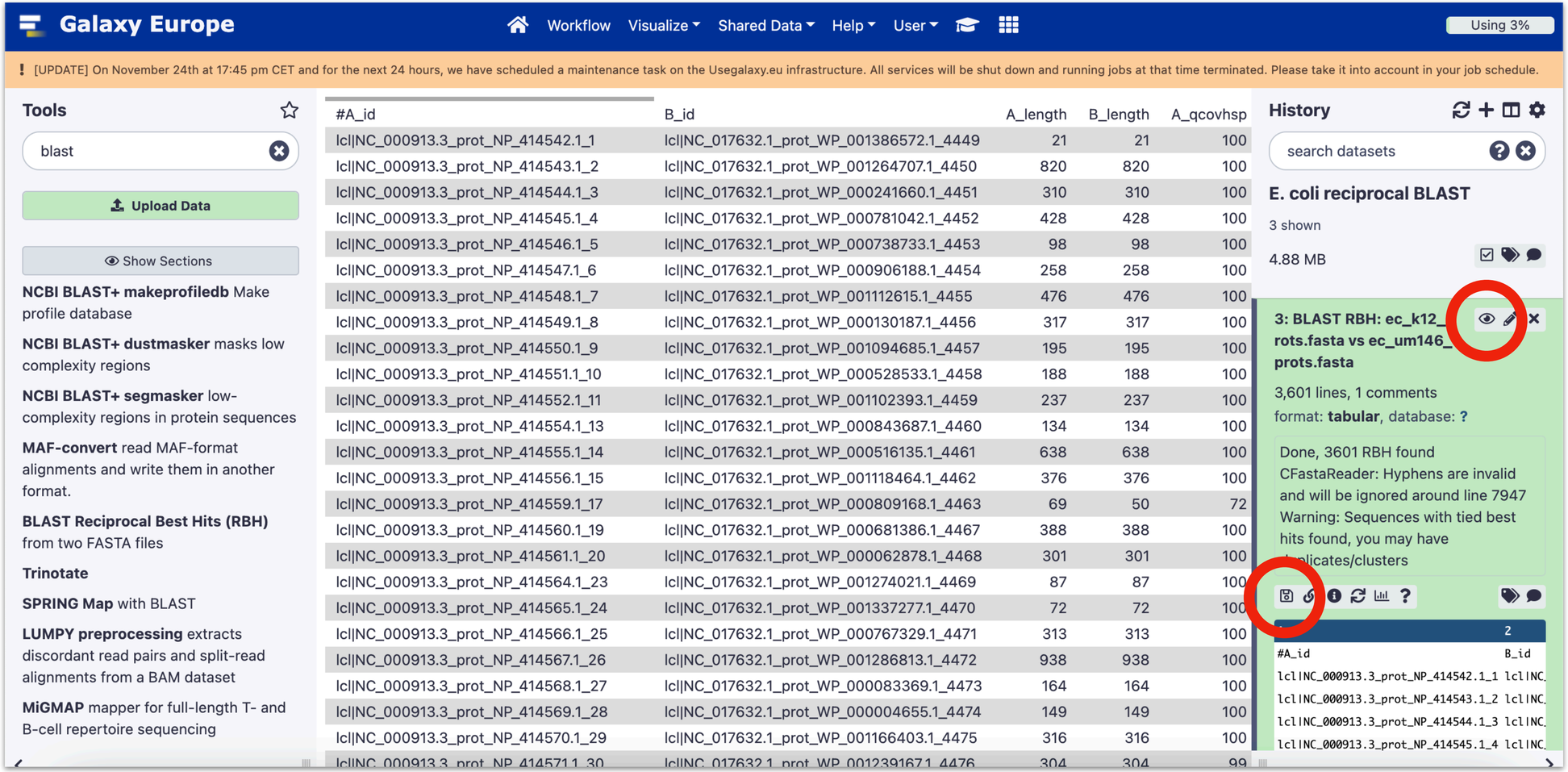
Select your two fasta files, change the "Molecule type" to "protein", and "blastp", then click "Execute"



Wait for the reciprocal BLAST task to turn from orange to green when it's complete



Have a look at your results, then click the little disk icon to download and save the file



Things to think about

- Check the **number of sequences** in both your "known" fasta file, and the "unknown" fasta file. You can find this by clicking the fasta filename in the history bar at the side to expand it.
- Check the number of lines in your BLAST results file the same way. The number of lines is the number of hits the "known" sequences had against the "unknown" ones.
- What proportion have hits?
- Do the hits have high percentage identity and coverage?
- What would have happened if we'd changed the default settings for the identity and coverage before running the reciprocal BLAST?

(the defaults (>70% identity, >50% coverage) are a good rule of thumb)

The perennial bioinformatics problem: IDs!

- Our fasta sequences look like this:

```
>lcl|NC_000913.3_prot_NP_414579.4_37 [gene=caiC] [locus_tag=b0037] ...
MDIIGGQHLRQMWDDLADVYGHKTALICESSGGVVNRYSYLELNQEINRTANLFYTLGIRKGDKVALHLD
NCPEFIFCWFGLAKIGAIMVPINARLLCEESAWILQNSQACLLVTSAQFYPMYQQIQQEDATQLRHICLT
DVALPADDGVSSFTQLKNQQPATLCYAPPLSTDDTAEILFTSGTTSRPKGVVITHYNLRFAGYYSAWQCA
LRDDDVYLTVMPAFHIDCQCTAAMAAFSAGATFVLVEKYSARAFWGQVQKYRATVTECIPMMIRTLMVQP
PSANDQQHRLREVMFYLNLSEQEKDAFCERFGVRLLTSYGMTETIVGIIGDRPGDKRRWPSIGRVGFCYE
AEIRDDHNRPLPAGEIGEICIKGIPGKTIFKEYFLNPQATAKVLEADGWLHTGDTGYRDEEDFFYFVDRR
CNMIKRGGENVSCVELENIIAAHPKIQDIVVVGIKDSIRDEAIKAFVVLNEGETLSEEEFFRFCEQNMAK
FKVPSYLEIRKDLPRNCSGKIIRKNLK
```

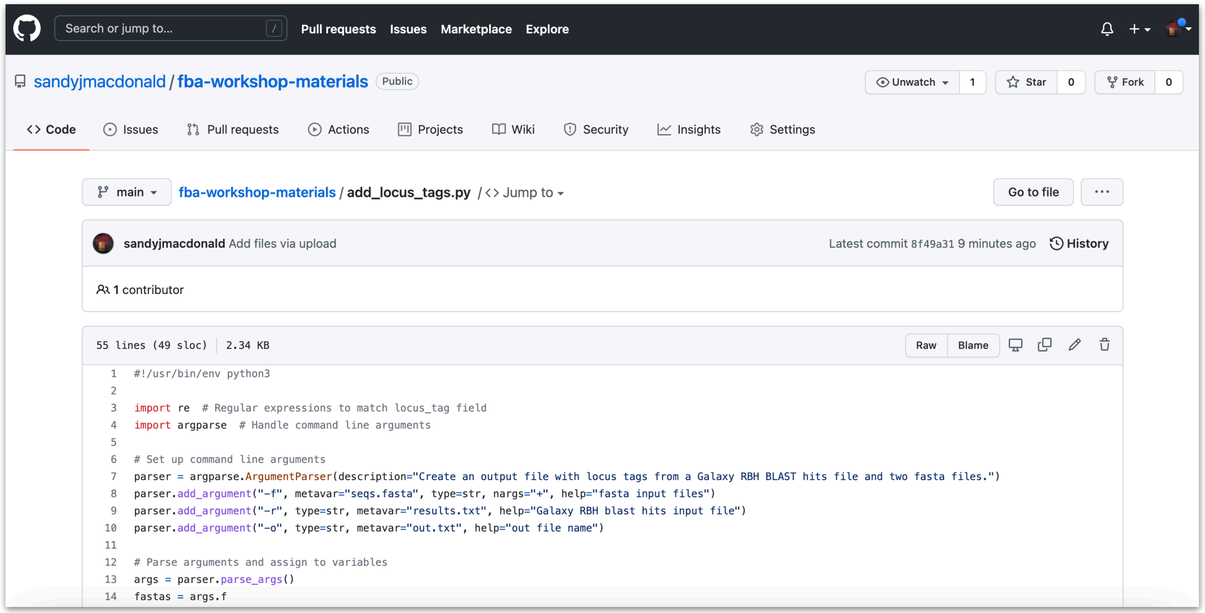
- But the gene reaction rules in the model look like this:

```
{"id": "CRNDCAL2",
"name": "D-Carnitine-CoA Ligase",
"metabolites":
{
    "adp_c": 1.0,
    "atp_c": -1.0,
    "coa_c": -1.0,
    "crnDcoa_c": 1.0,
    "crn__D_c": -1.0,
    "pi_c": 1.0
},
"lower_bound": 0.0,
"upper_bound": 999999.0,
"gene_reaction_rule": "b0037",
"subsystem": "Oxidative Phosphorylation"},
```

Python to the rescue!

My `add_locus_tags.py` script

- I've written a Python script that takes the two fasta files, the blast results file, and the name you want to call your output file, and it creates an output file with the matching locus tags on each line
- You can find it here: https://github.com/sandyjmacdonald/fba-workshop-materials/blob/main/add_locus_tags.py
- Put the Python script in the same directory as your two fasta files and blast results file and then run it by typing this in the terminal:




```
python3 add_locus_tags.py -f known.fasta unknown.fasta -r blast_results.tabular -o results_locus_tags.txt
```

Filtering your model

- With our list of locus tags that match from our "known" genome against our "unknown" genome, we can now go through the reactions in the "known" model and check whether the genes in the gene reaction rule have matches
- What's the best way to do this filtering? Python again!
- **COBRApy** is a Python library for working with FBA models: https://github.com/opencobra/cobrapy
- We can use it to check our list of locus tags against each reaction, keep only the matching ones, and then swap the locus tags, replacing the "known" locus tags with the "unknown" (now also known!) ones

Python to the rescue again!
My `filter_fba_model.py` script

- To make the process a bit easier, I've made a Python script that uses COBRApy to filter an FBA model in json format if you give it a text file. like the one we created, that maps the known locus tags to the unknown ones
- You can find it in the same GitHub repository: https://github.com/sandyjmacdonald/fba-workshop-materials/blob/main/filter_fba_model.py
- Run it by typing, for example:

```
Model summary:

1138 genes matched from 1261 genes in original model (90%)

Missing genes:
b0423,b3629,b1800,b0731,b2538,b0341,b2092,b2542,b0616,b1907,b2484,b1296,b2485,b1385,b1363,b0351,b3627,b1488,b3628,b0349,b2027,b4177,b4474,b3579,b0615,b4291,b2487,b3213,b3875,b0353,b2530,b1297,b1302,b3374,b2036,b2541,b2920,b4287,b2203,b0333,b3990,b0260,b1483,b2486,b3991,b2482,b3577,b3578,b0732,b2481,b1484,b0614,b3502,b2038,b4031,b0584,b0150,b2037,b0340,b3625,b4301,b0339,b3622,b3648,b2539,b2492,b2205,b0261,b2490,b3626,b4321,b2204,b4288,b0650,b3927,b3212,b0352,b0617,b3623,b3624,b2094,b2540,b2489,b2393,b2034,b2202,b1102,b3992,s0001,b1486,b1398,b2488,b1298,b4407,b4289,b2035,b2917,b1801,b1485,b2093,b4120,b3370,b1386,b0348,b0273,b1301,b0070,b4356,b1300,b0347,b1778,b1006,b2483,b3371,b3966,b1487,b2919,b4290,b2032,b2010,b0350,b2033,b2206

2266 reactions added from 2382 reactions in original model (95%)

Missing reactions:
42A12BOOXpp,ACALDtpp,ADSS,ACONIs,ALAALAD,ADOCBLtonex,ALDD19xr,ALDD2x,AOBUTDs,AS03t8pp,ARBTNexs,ATPHs,CBItonex,CBL1tonex,CINNDO,CITL,CO2tpp,CPGNexs,CYNTAH,CYNTt2pp,CPGNtonex,DATPHs,3HCINNMH,3HPPPNH,DGK1,DHCIND,DHCINDO,DHPPD,DHPTDCs,ECA40ALpp,FE3DCITabcpp,FE3DCITtonex,FALDtpp,FALGTHLs,FE3HOXexs,FE3HOXtonex,FECRMexs,FECRMtonex,FRULYSDG,FRULYSE,FRULYSK,FRULYSt2pp,FEENTERexs,FEENTERtonex,FRUURt2rpp,FEOXAMexs,FEOXAMtonex,GALCTNLt2pp,GALT1,GALTptspp,G5SADs,GGGABADr,GGGABAH,GGPTRCO,GGPTRCS,GK1,GLUSy,GLCTR2,GLCTR3,GLYALDtpp,GLYBt2pp,GLYCtpp,HCINNMt2rpp,GTPHs,HCYSMT,HCYSMT2,HEPK2,HEPT4,H2St1pp,H2tpp,HKNDDH,HKNTDH,HOPNTAL,HPPPNDO,HPPPNt2rpp,MALDDH,MANGLYCptspp,MANPGH,MELIBt2pp,MCITS,MMCD,METOX1s,METOX2s,MMETt2pp,METSOXR2,MMM2,NO3R1bpp,NO3R2bpp,NOtpp,O16A4Lpp,O16AP1pp,O16AP2pp,O16AP3pp,O16AT,O16AUNDtpp,O16GALFT,O16GLCT1,PACCOAL,MOAT3C,O2tpp,OP4ENH,PEAMNOpp,PPCSCT,N2Otpp,PPPNDO,PSCLYSt2pp,RHAT1,SO2tpp,THZPSN,TDPDRE,URAt2pp_copy2,UREAtpp,TYROXDApp,XYLUt2pp,XYLt2pp,UDPGALM

Filtered model written to example_files/filtered_iAF1260.json
```

```
python3 filter_fba_model.py -l locus_tags_map.txt -m model.json
```

You're only just at the start!

- There's still *a lot* left to do!
- Your model will likely have a lot of gaps in it (and errors)
- Gap-filling, which COBRApy can do, will give you a working model but ideally you need to find candidate genes for those gap-filled reactions
  - **Try re-running the BLAST with lower cutoffs to see if any of these missing genes turn up. Promiscuous enzymes? Side reactions?**
- Are all the assignments from our BLAST correct?
  - **Check if the genes in the "unknown" genome have been characterised.**
- Do any exchange reactions need to be added to make pathways work?
- What should the biomass reaction be?

Give it a go...

- Run through this process and pick your own "unknown" strain or species
- I'd recommend starting with another strain of the same species and if your new model is too good then try a different species
- Does your model work at all?
- If not, then can you try gap-filling?

Key things

- What "unknown" organism did you pick, and why?
- What proportion of the "known" genes had hits to "unknown" genes?
- Pick another set of BLAST cutoffs and see how this changes
- What proportion of genes and reactions are retained in your filtered model compared to the original model?
- Can you think of reasons why genes could be missing?

Questions? Queries?

Pop me an email with any questions or queries:
s.macdonald@york.ac.uk