RPTU Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

Unleashing the Analytical Power of F#: Empowering Biotechnological Data Science Education and Research

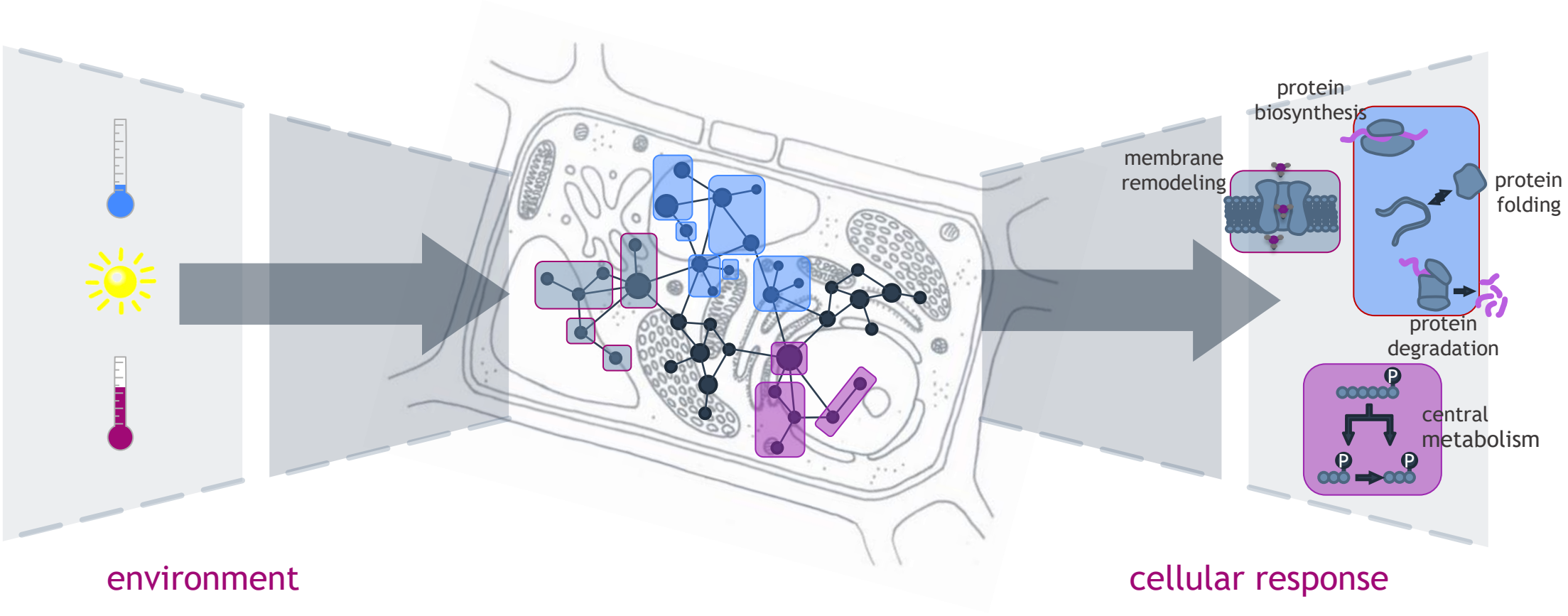
Timo Mühlhaus

CSB

COMPUTATIONAL SYSTEMS BIOLOGY

Data Science in F#
Berlin, 28–30 September 2023

Computational Systems Biology



Computational Systems Biology Group

TRR 175

SPONSORED BY THE
Federal Ministry of Education and Research

nfdi Nationale Forschungsdaten Infrastruktur

bioComp

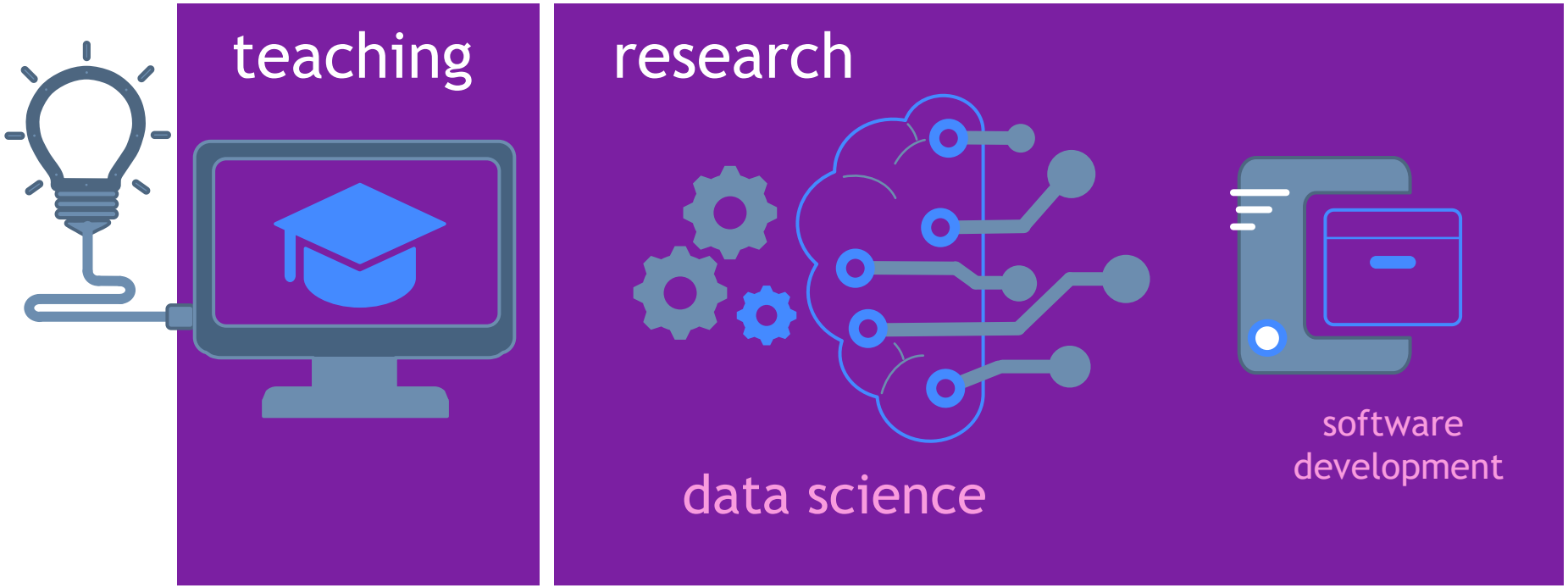
Deutsche Forschungsgemeinschaft DFG

Computational Systems Biology at RPRU

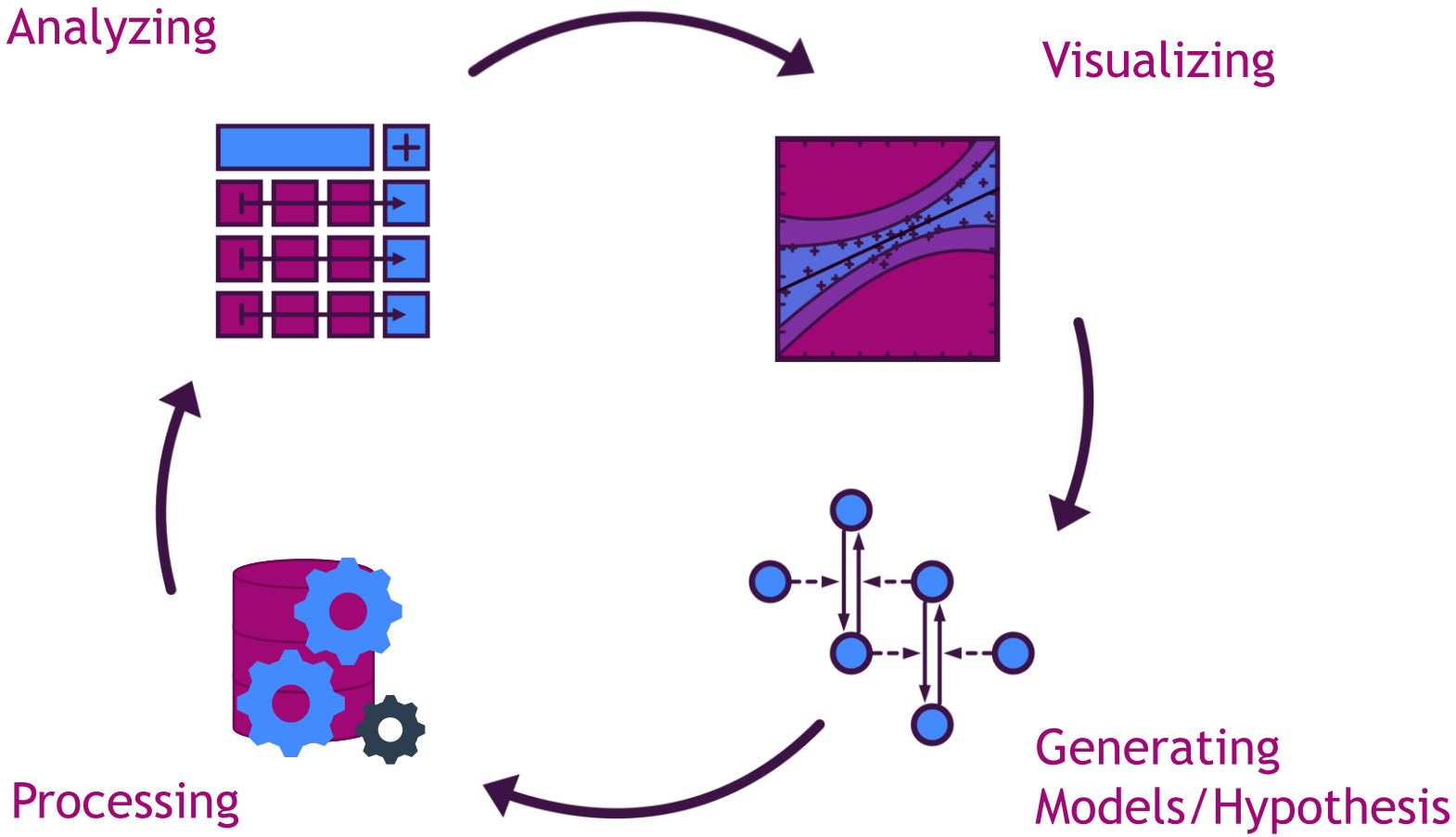
TU Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

RP

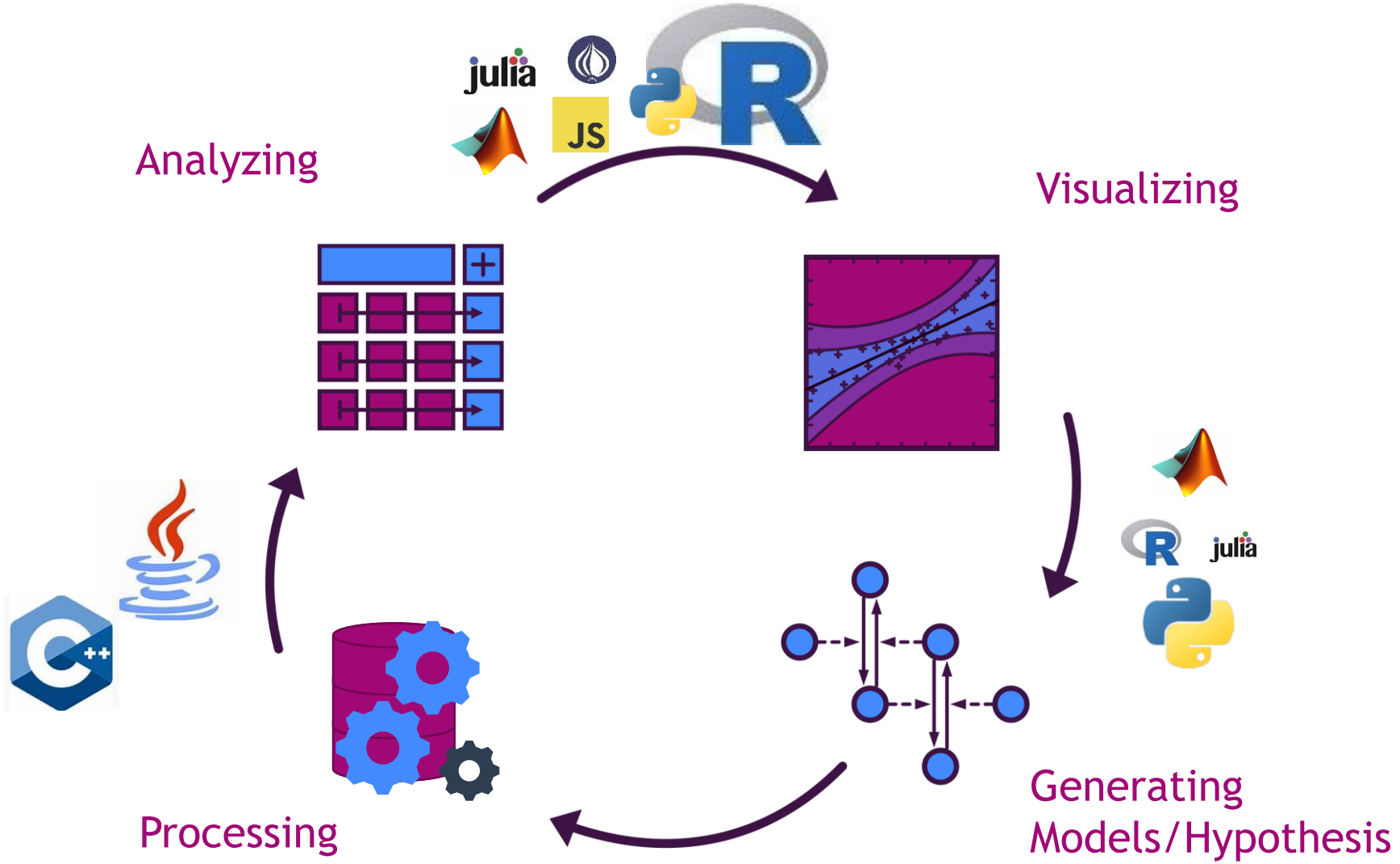
Tasks in Academia



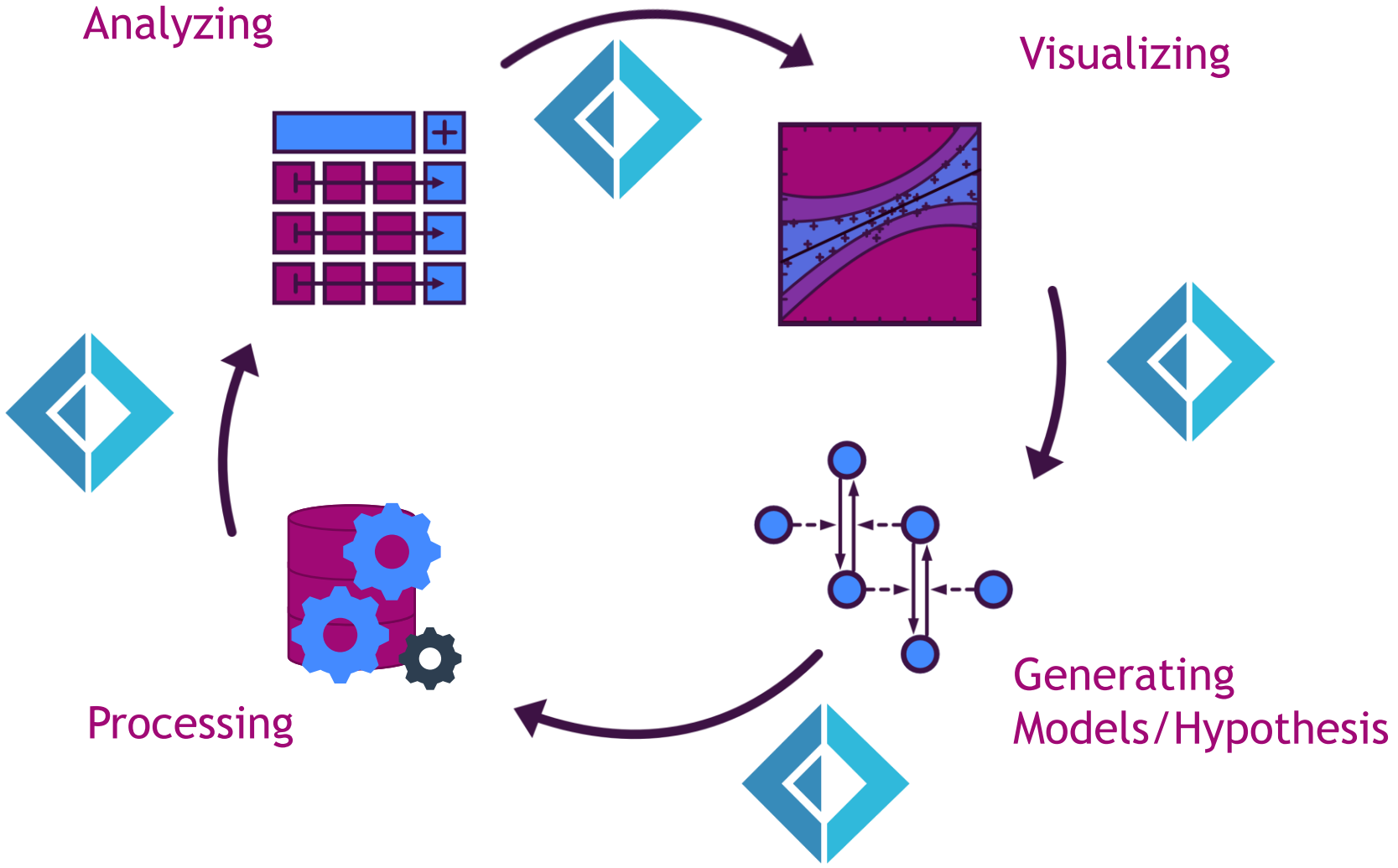
Data Science in Biology



Data Science in Biology

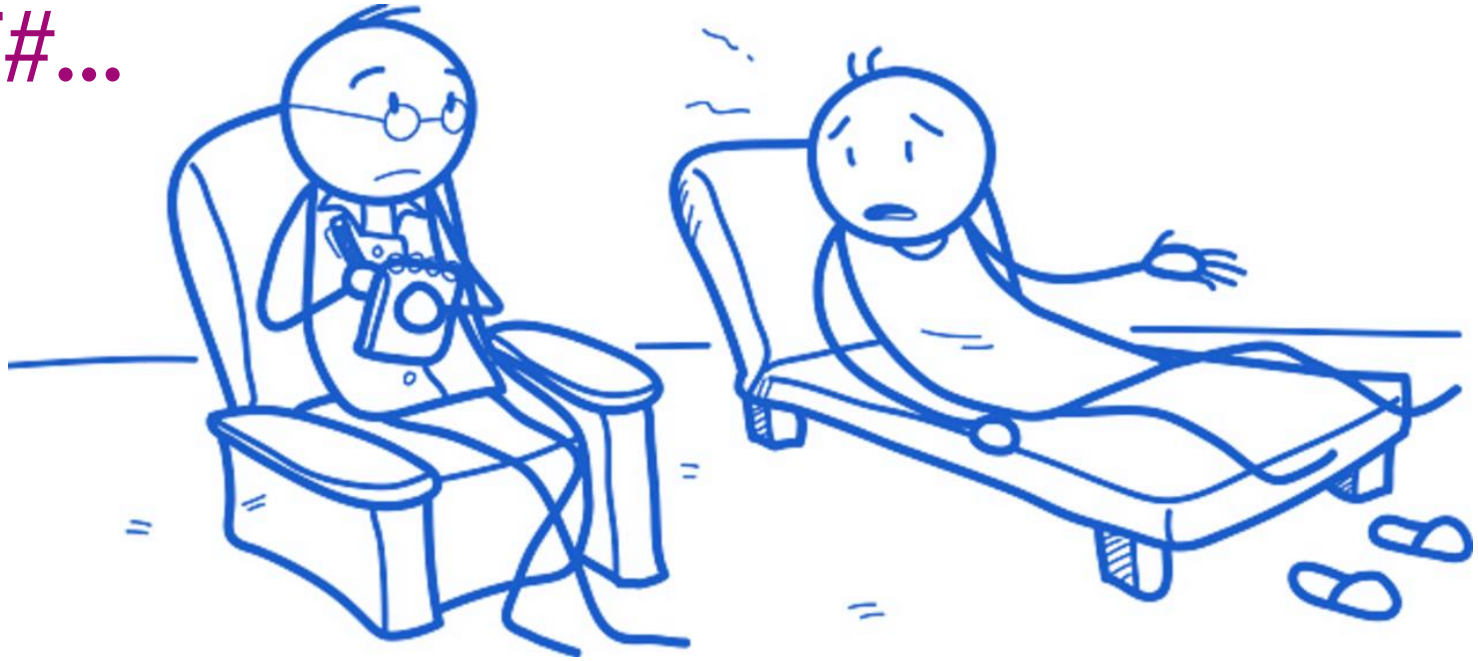


F# to Homogenize the ‘Zoo’ of Languages



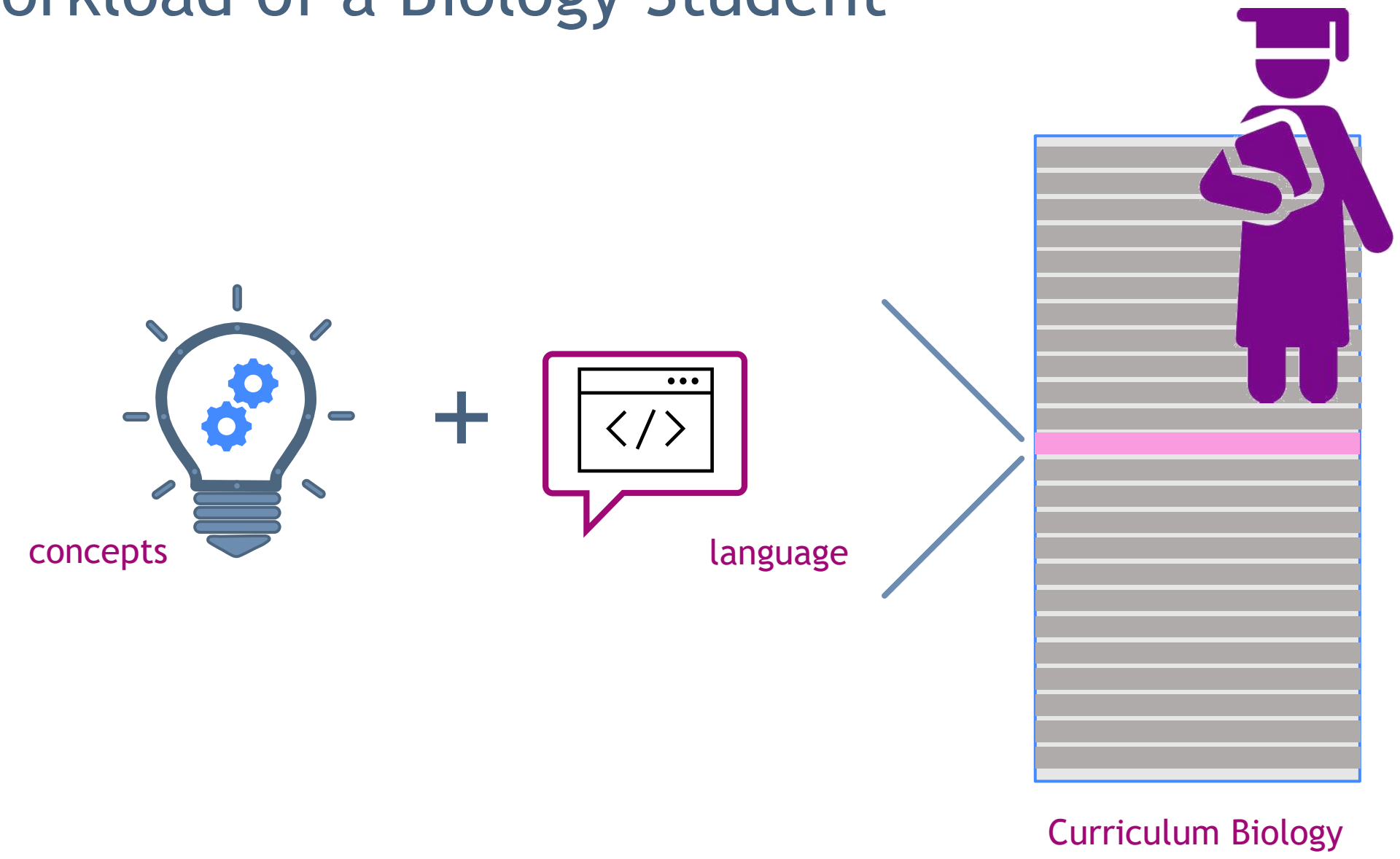
Biggest Question of the Students in Biology

Why F#…



…when all my friends use python?

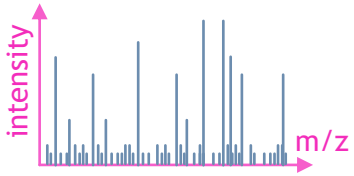
Workload of a Biology Student



...and of course ...

Take a look at a bioinformatics workflow

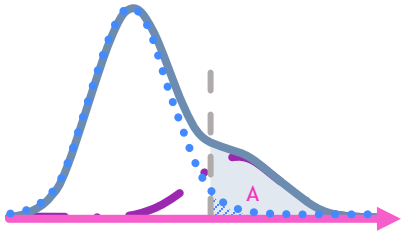
measurements



processing

P1	P2	P3	P4
0.1	0.3	0.1	0.2
0.5	0.2	0.1	0.1
0.2	0.2	0.6	0.5
0.2	0.3	0.2	0.2

statistical analysis



integration

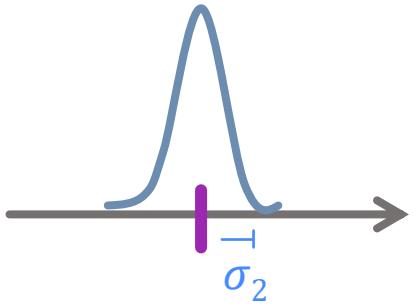
interpretation

AGGTG AGAGG
CAGAG
GGTGA
GAGGT
CAGAGGTGA

processing

CAGAGGTGA
CAGAG
AGAGG
GAGGT
AGGTG
GGTGA

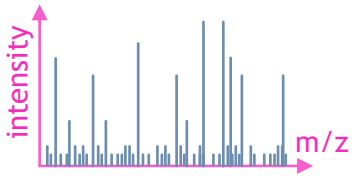
statistical analysis



measurements

Take a look at a bioinformatics workflow

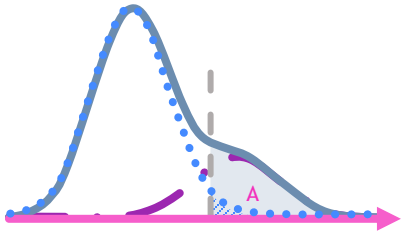
measurements



processing

P1	P2	P3	P4
0.1	0.3	0.1	0.2
0.5	0.2	0.1	0.1
0.2	0.2	0.6	0.5
0.2	0.3	0.2	0.2

statistical analysis



integration

interpretation

---

information

function(s)

information

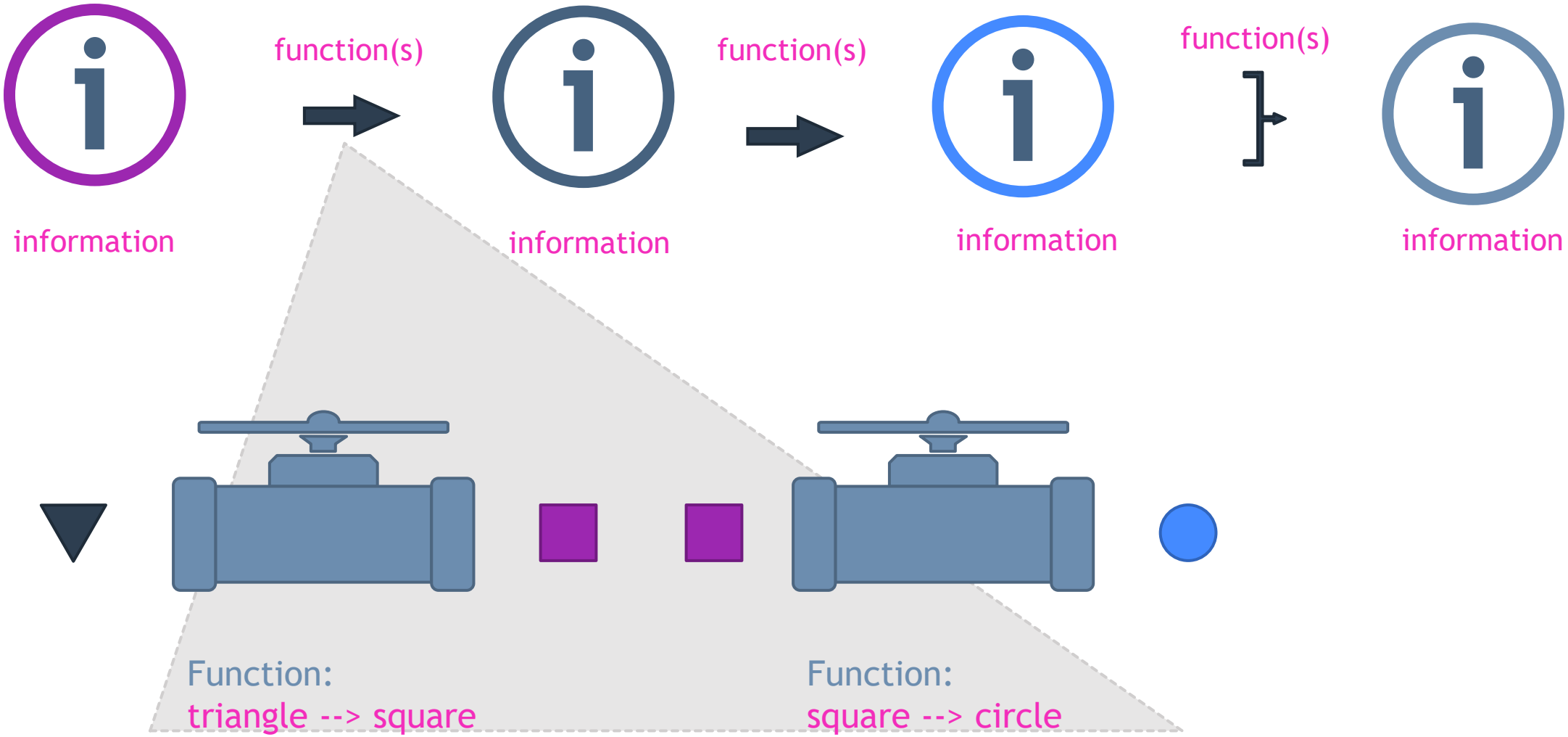
function(s)

information

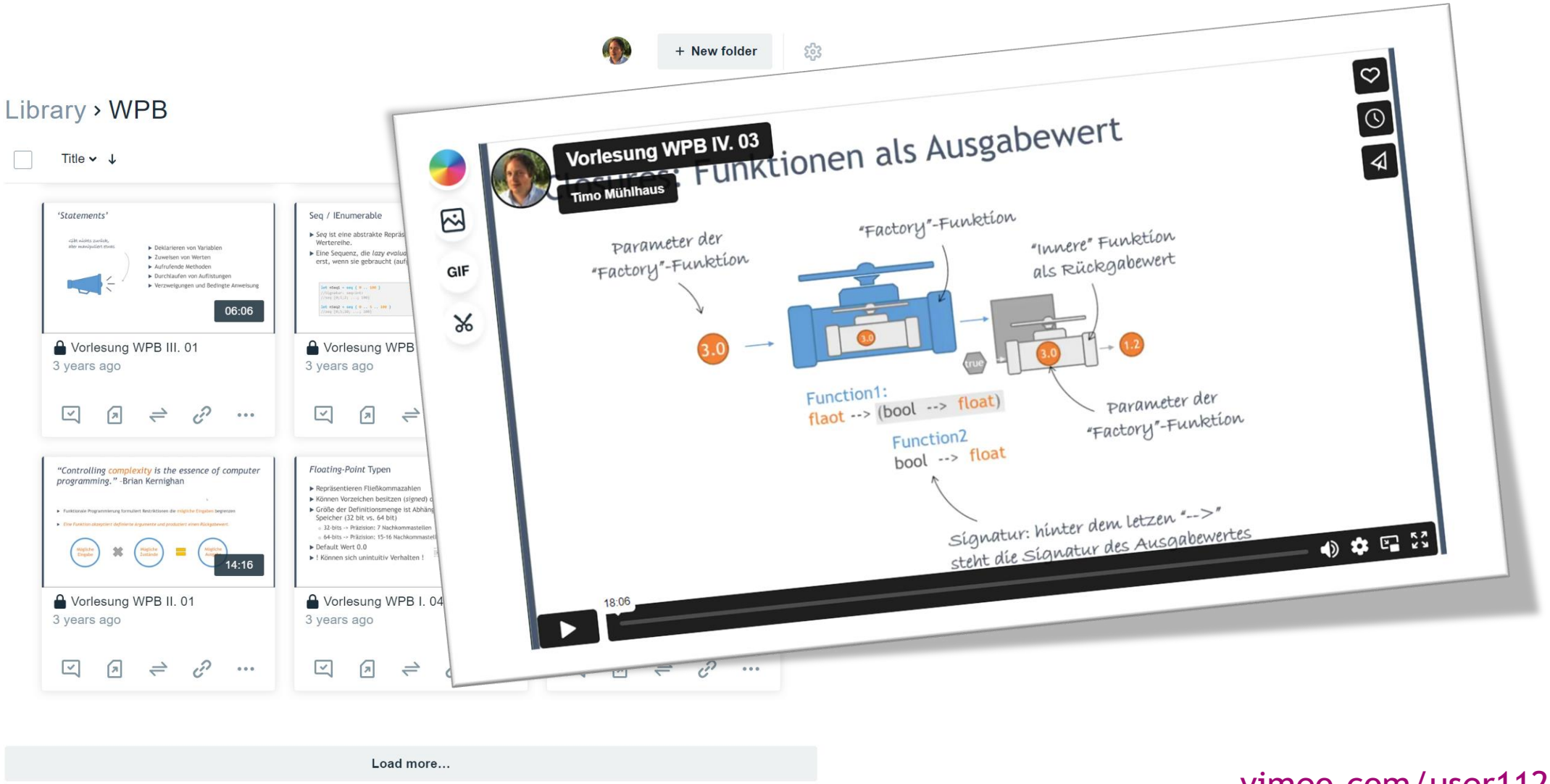
function(s)

information

Function Driven Transformation

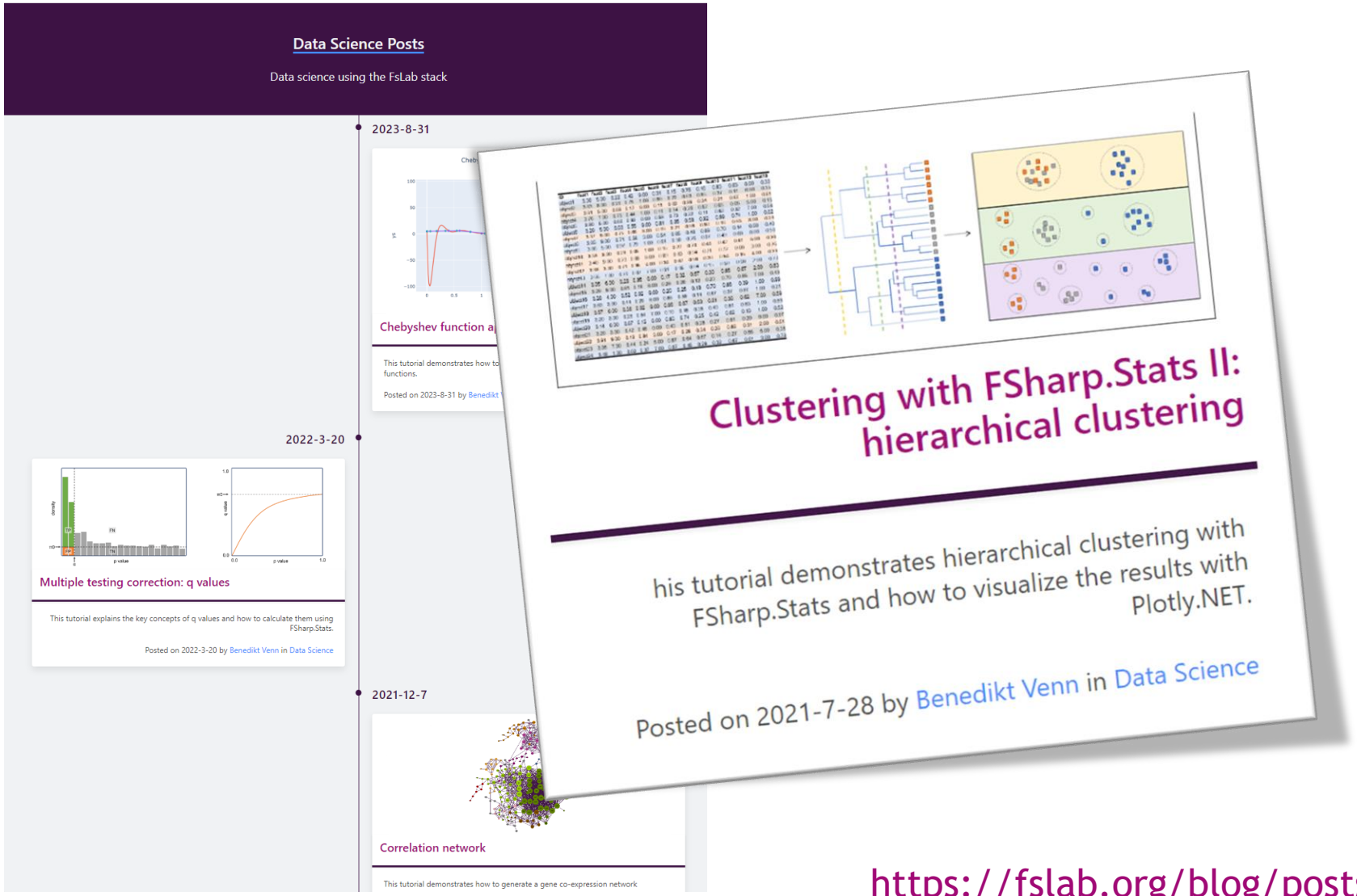


Lecture Series F# for Biologists



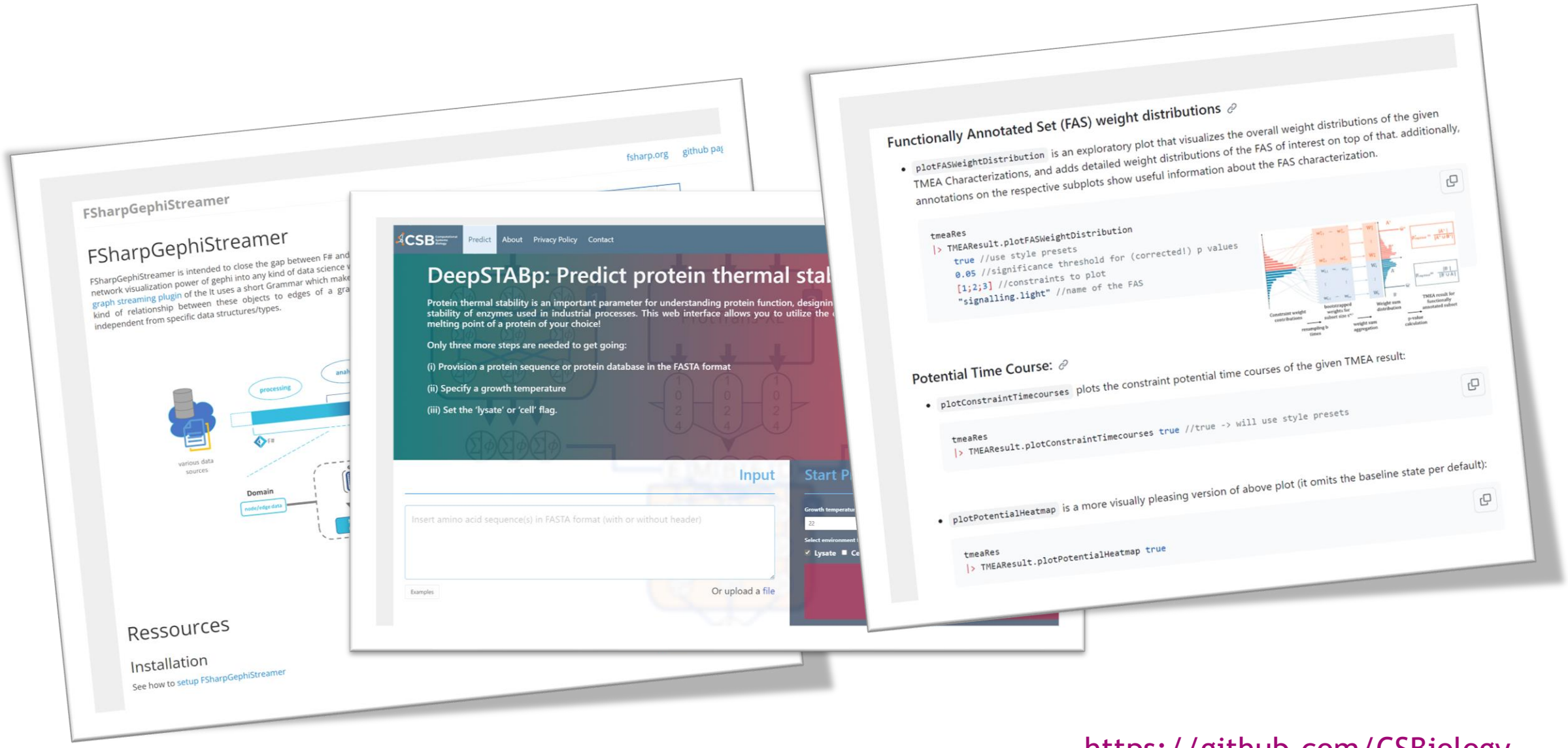
vimeo.com/user112715137/

Blog Posts as Resources for Students



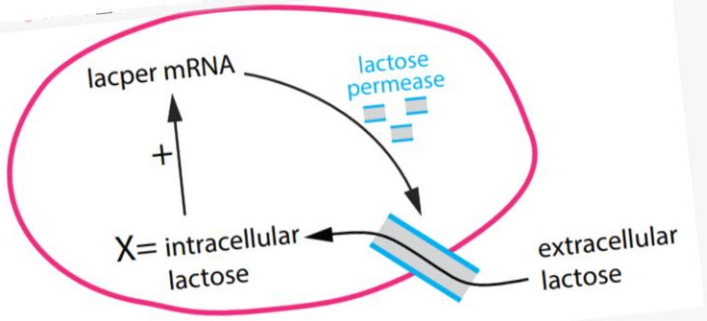
https://fslab.org/blog/posts/categories/Datascience.html

Student Projects



https://github.com/CSBiology

Notebooks



Die Differenzialgleichung für die Konzentration von intrazellulärer Lactose kann somit folgendermaßen beschrieben werden. Es gibt einen unregulierten geringen Lactoseimpo
außerdem einen regulierten Lactoseimport, dieser wird β genannt analog zur Proteinproduktion von $\beta - \text{Galactosid-Permease}$ die nötig ist damit dieser regulierte Transp
intrazelluläre Lactose zu mehr Import führt, kann hier von einer positiven Autoregulation gesprochen werden. Abschließend wird Lactose natürlich auch von der Zelle verbrauc
die genutzt wird wieder von der Lactosekonzentration abhängt. Die Differenzialgleichung ist somit

$$Lactosekonzentration'_{intrazellulär} = \frac{dLactosekonzentration_{intrazellulär}}{dt} = \alpha + \beta * \frac{Lactosekonzentration_{intrazellulär}}{K_D + Lactosekonzentration_{intrazellulär}} - \gamma * Lactosekonzent$$

Da die Simulationen nur auf die richtige Differenzialgleichung achten, aber nicht auf die genutzten Namen der Parameter, verkürzen wir die Gleichung zu:

$$inlac' = \frac{dinlac}{dt} = \alpha + \beta * \frac{inlac}{K_D + inlac} - \gamma * inlac$$

Aufgabe 4.1

***Setzen Sie die Gleichung für die intrazelluläre Lactosekonzentration in die nachfolgende Simulation ein (Bitte beachten Sie, dass Sie die Lactosekonzentration a
müssen, ohne "". Beachten Sie ebenfalls das α, β und γ als "alpha", "beta" und "gamma", ohne "", eingesetzt werden müssen und nicht als Symbole, und das
wird).***

```
//Konstanten
let alpha =  0.5  // Konstante für die Zunahme der Lactosekonzentration (unreguliert)
let beta  =  10.0 // Konstante für die Zunahme der Lactosekonzentration (reguliert)
let gamma =  1.0  // Konstante für die Verringerung der Lactosekonzentration
let K_d =    5.0  // Dissoziationskonstante

// unsere DGL als  Modell (Model)
let dP_dt : Model =
    fun P t ->
```

```
let berechneEuler x0 y0 f h n =
    [|x0 .. h .. x0 + (n / (1. / h))|]
    |> Array.scan (
        fun acc x ->
            acc + h * (f acc)
    ) y0

berechneEuler t0 y0 (fun t -> r * t) h n
```

fsharp - F# Script Code

Hier das Ganze als Diagramm visualisiert:

+ Code + Markdown

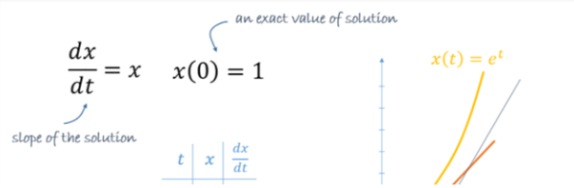
```
// eine Zwischenspeicherung unseres Ergebnisses:
let eulerYWerte = berechneEuler t0 y0 (fun t -> r * t) h n

eulerYWerte
|> Array.indexed
|> Chart.Point
|> Chart.withTitle "Euler-Verfahren, Schrittweite = 1, Anzahl Punkte = 10"
|> Chart.withXAxisStyle "t [min]"
|> Chart.withYAxisStyle "Bakteriendichte"
```

fsharp - F# Script Code

Aufgabe 1.3: Wie könnten Sie als Biolog:in einen solchen Startpunkt P experimentell bestimmen?

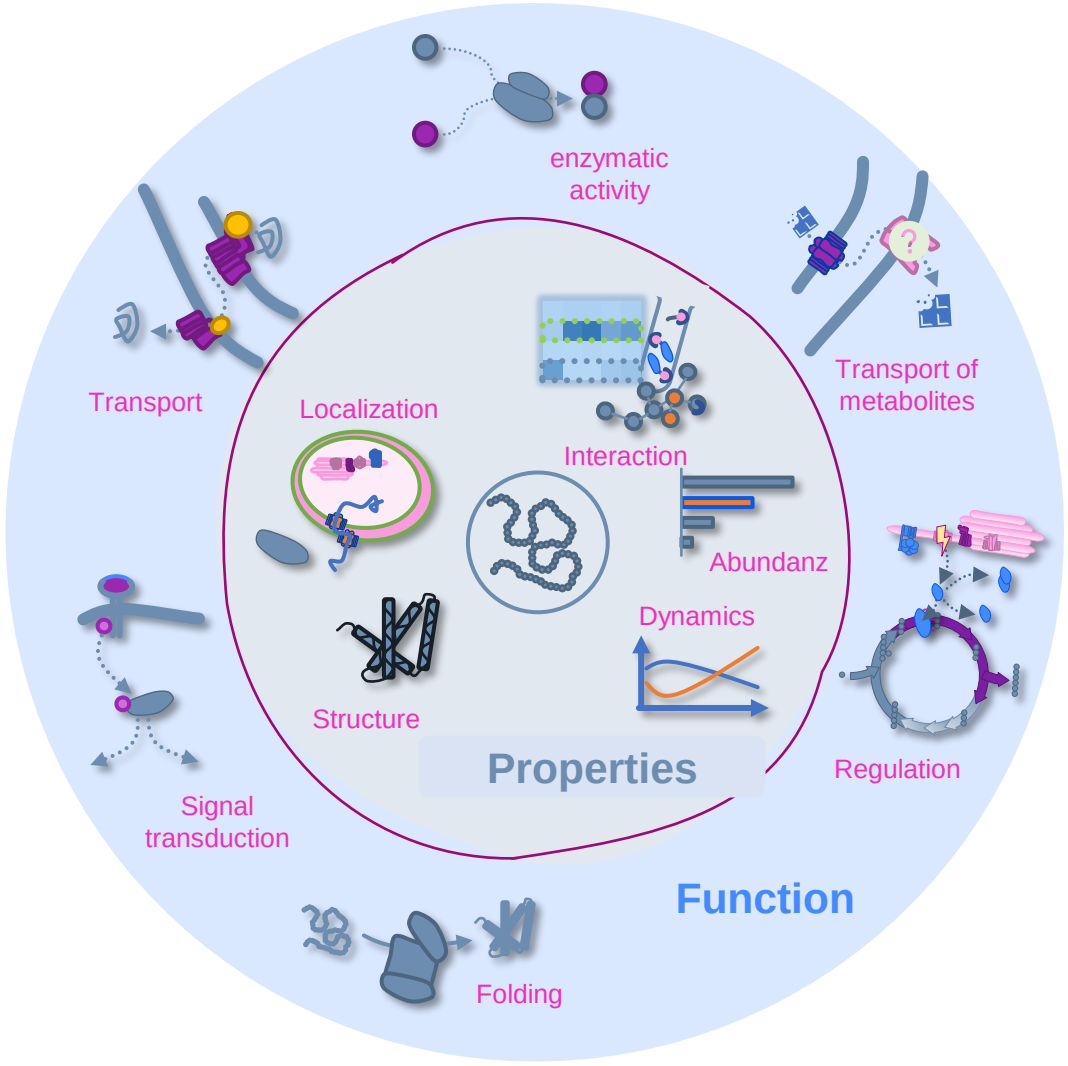
Antwort:

Betrachten Sie folgende Abbildung aus der Vorlesung:



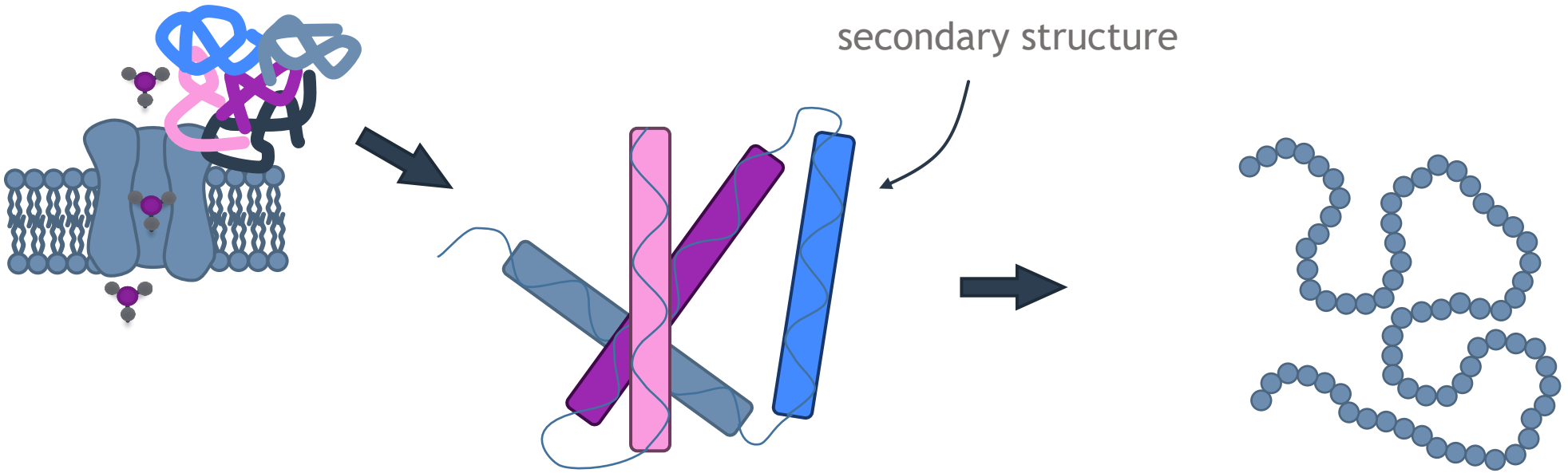
https://github.com/CSBiology/BIO-MBP-06-U-2

Proteins: Key Players in Living Systems



- Proteins are crucial for the **biotechnological production** of high-quality biomaterials
- Proteins are important targets for the **diagnosis and treatment** of diseases caused by protein malfunctions

All Information are Encoded in Biological Sequences



tertiary structure

primary structure

=

sequence
of amino acids

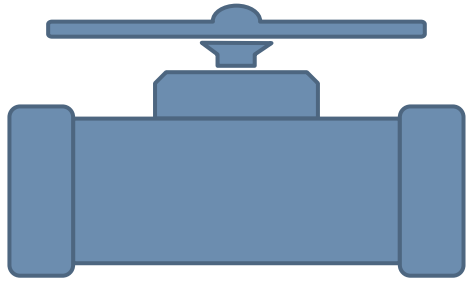
Working with Biological Sequences

- The representation of biological sequences is basically a sequence of characters.

Working with Biological Sequences

"AGTACACTGGT"

ambiguous



strongly typed

BioArray.ofAminoAcidString
string --> BioArray.BioArray<AminoAcid>

```
BioArray.ofAminoAcidString "AGTACACTGGT"
```

F#

```
val it : BioArray.BioArray<AminoAcid> =
[|Ala; Gly; Thr; Ala; Cys; ...|]
```

BioFSharp – Computational Biology in F#

BioFSharp fsharp.org github page

BioFSharp

BioFSharp aims to be a user-friendly library for Bioinformatics written in F#. It contains the basic data structures for common biological objects like amino acids and nucleotides based on chemical formulas and chemical elements.

Example

This example demonstrates using a function defined in BioFSharp library.

```
1: #r "BioFSharp.dll"
2: open BioFSharp
3:
4: /// Creates a BioSeq of the given peptide string
5: BioSeq.ofAminoAcidString   "REYAHMIGMEYDTVQK"
6:
7: /// Creates a BioArray of the given peptide string
8: BioArray.ofAminoAcidString "REYAHMIGMEYDTVQK"
9:
10: /// Creates a BioList of the given peptide string
11: BioList.ofAminoAcidString  "REYAHMIGMEYDTVQK"
```

Samples & documentation

The library comes with comprehensible documentation. It can include tutorials automatically generated from `*.fsx` files in [the content folder][content]. The API reference is automatically generated from Markdown comments in the library implementation.

- Tutorial contains a further explanation of this sample library.
- API Reference contains automatically generated documentation for all types, modules and functions in the library. This includes additional brief samples on using most of the functions.

BIOFSHARP

Home page

Get Library via NuGet

Source Code on GitHub

License

Release Notes

GETTING STARTED

BioSequence

Spectrum centroidization

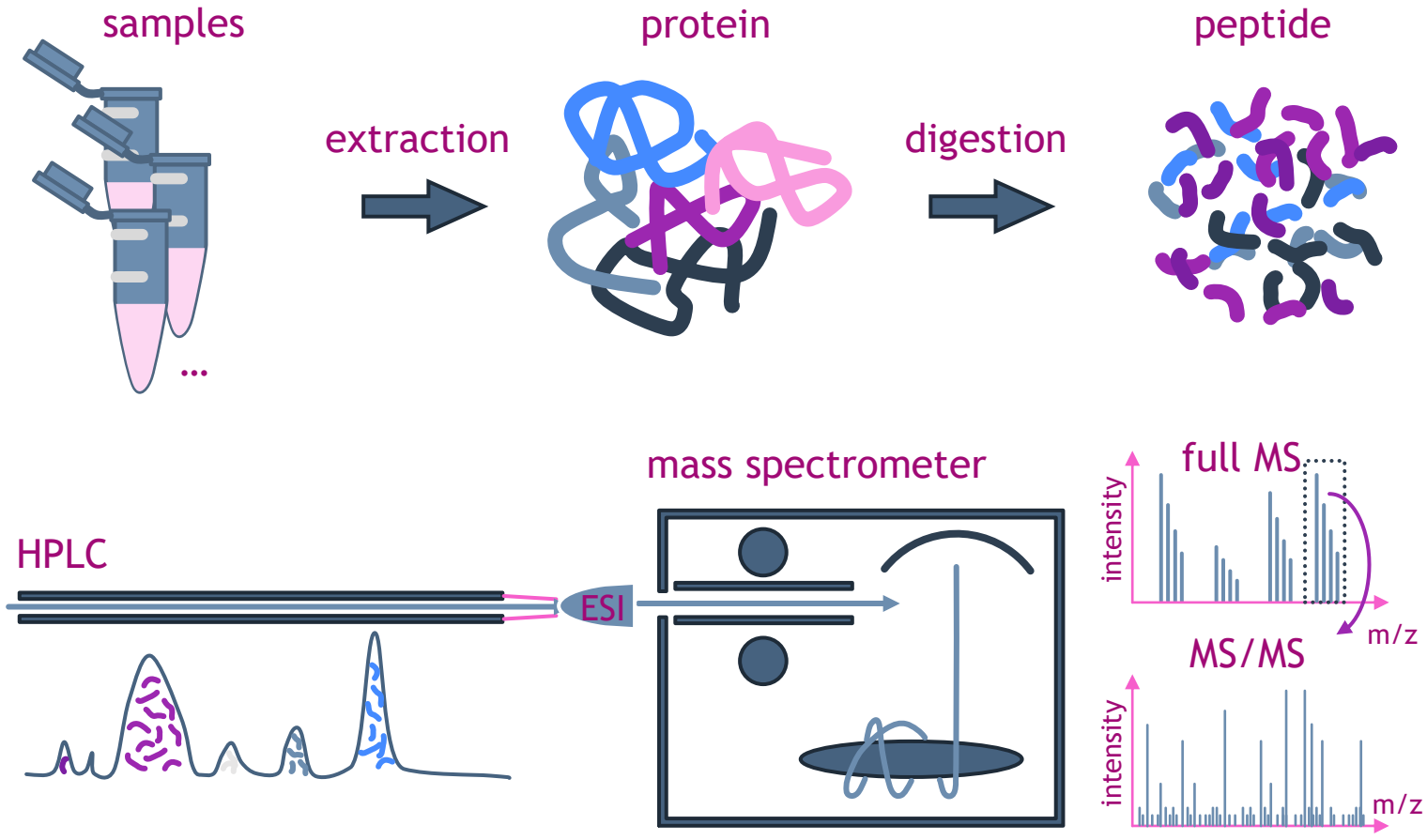
Charge state determination

Peptide look up

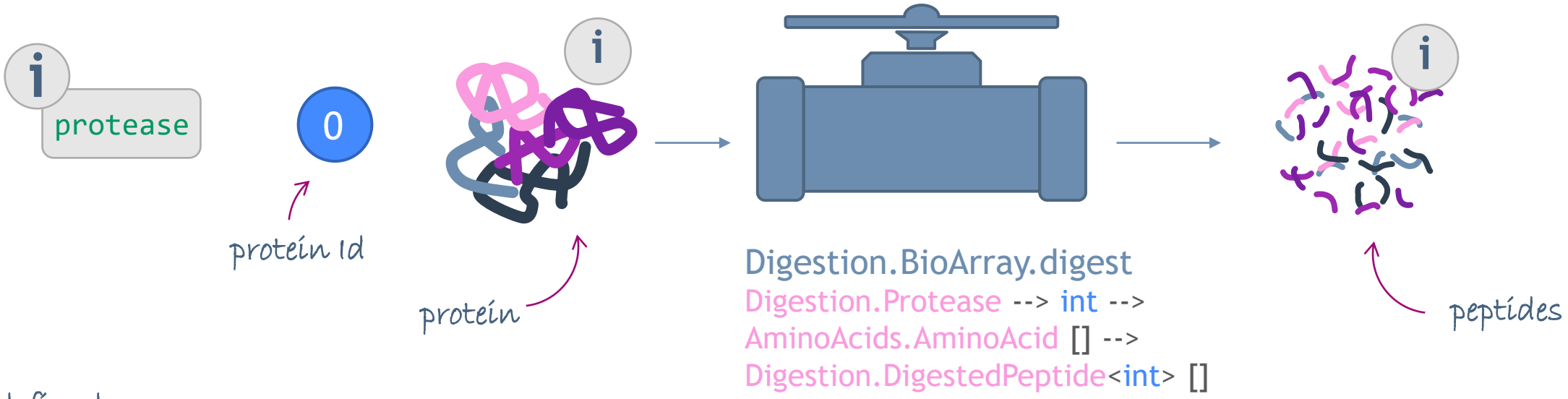
Peptide identification

https://github.com/CSBiology/BioFSharp

Proteomics workflow



Step1: Proteolytic Digestion of Proteins



Predefined
Protease
Trypsin

```
let trypsin = Digestion.Table.Trypsin
let digestedRBCS = rbcsProtein |> Digestion.BioArray.digest trypsin 0
```

F#

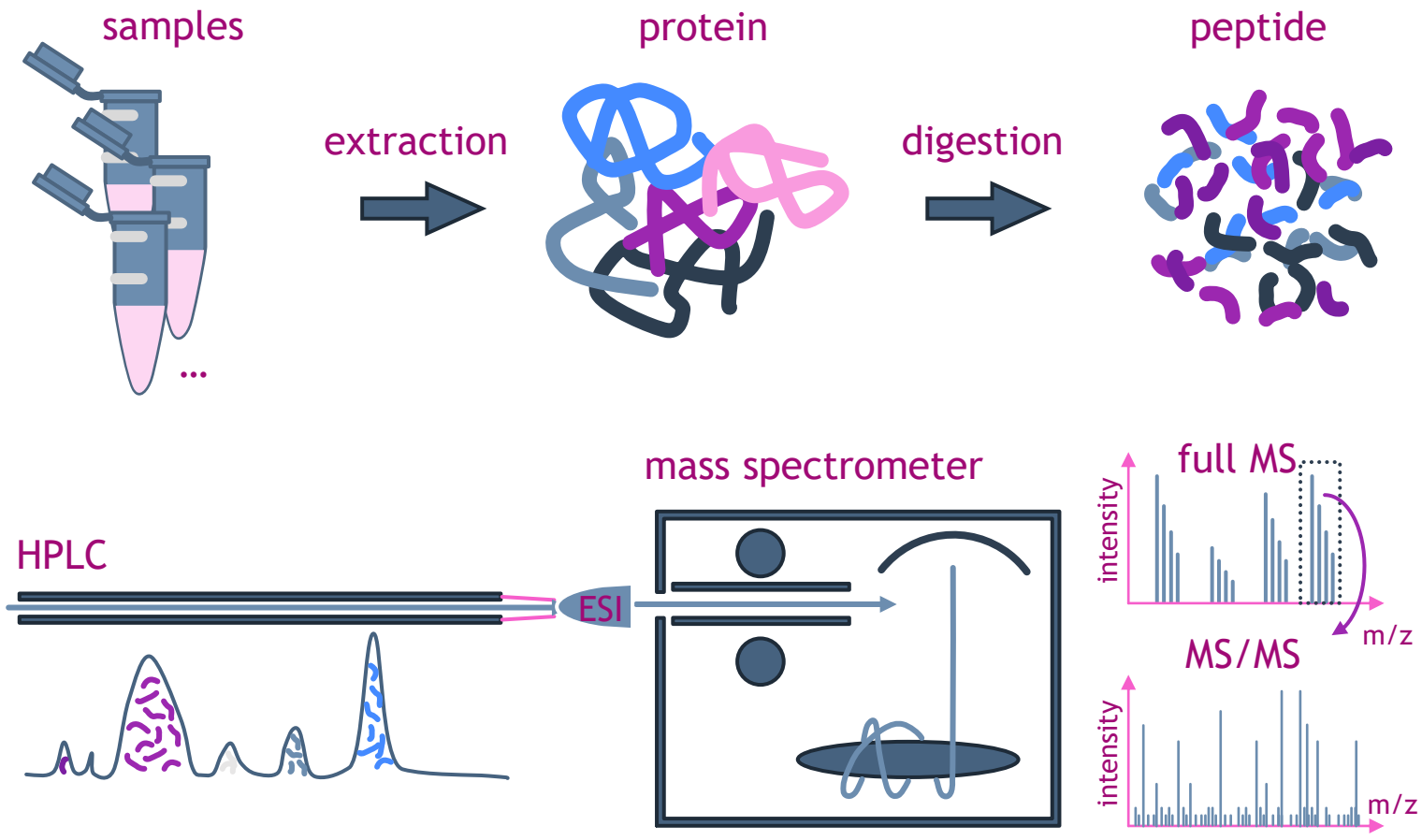
Proteolytic Digestion

```fsharp
let trypsin = Digestion.Table.Trypsin
let digestedRBCS = rbcsProtein |> Digestion.BioArray.digest trypsin 0
```

F#

```
val digestedRBCS : Digestion.DigestedPeptide<int> [] =
[|
    { ProteinID = 0
      MissCleavages = 0
      CleavageStart = 0
      CleavageEnd = 9
      PepSequence = [Met; Met; Val; Trp; Thr; Pro; Val; Asn; Asn; Lys]
    };
    ...
|]
```

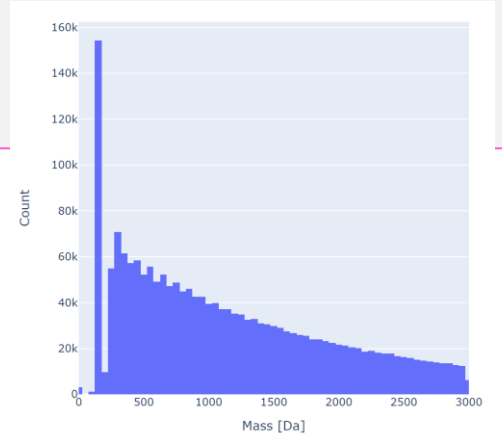
Proteomics Workflow



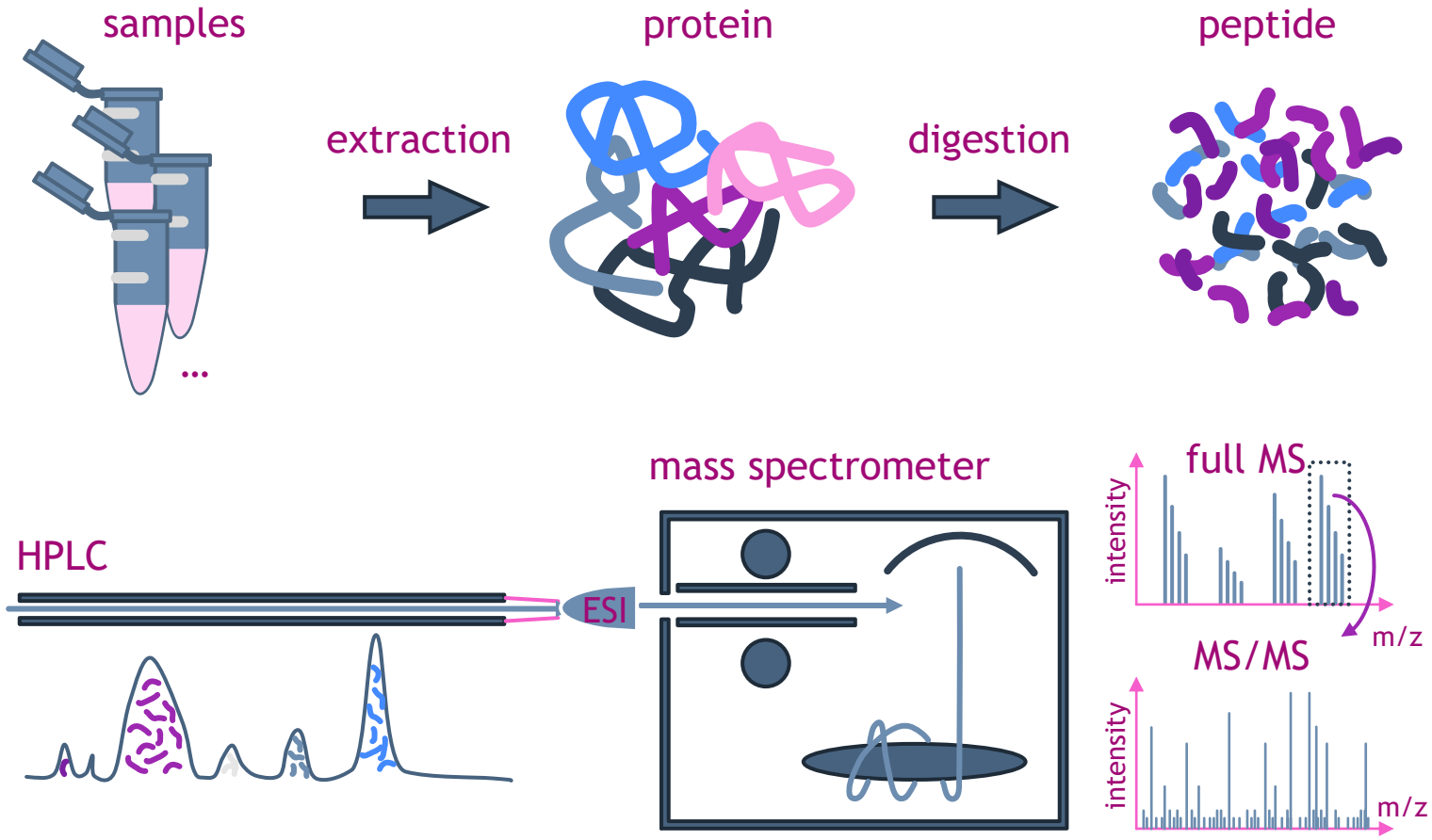
Calculating Peptide Masses

F#

```
digestedProteins
|> Array.map (fun peptide ->
     // calculate mass for each peptide
     BioSeq.toMonoisotopicMassWith
           (BioItem.monoisoMass ModificationInfo.Table.H2O) peptide
     )
|> Array.filter (fun x -> x < 3000.)
// visualize distribution of all peptide masses < 3000 Da
|> fun masses -> Chart.Histogram(data = masses, orientation = Vertical, NBinsX = 100)
|> Chart.withXAxisStyle (TitleText = "Mass [Da]", MinMax = (0., 3000.))
|> Chart.withYAxisStyle "Count“
```

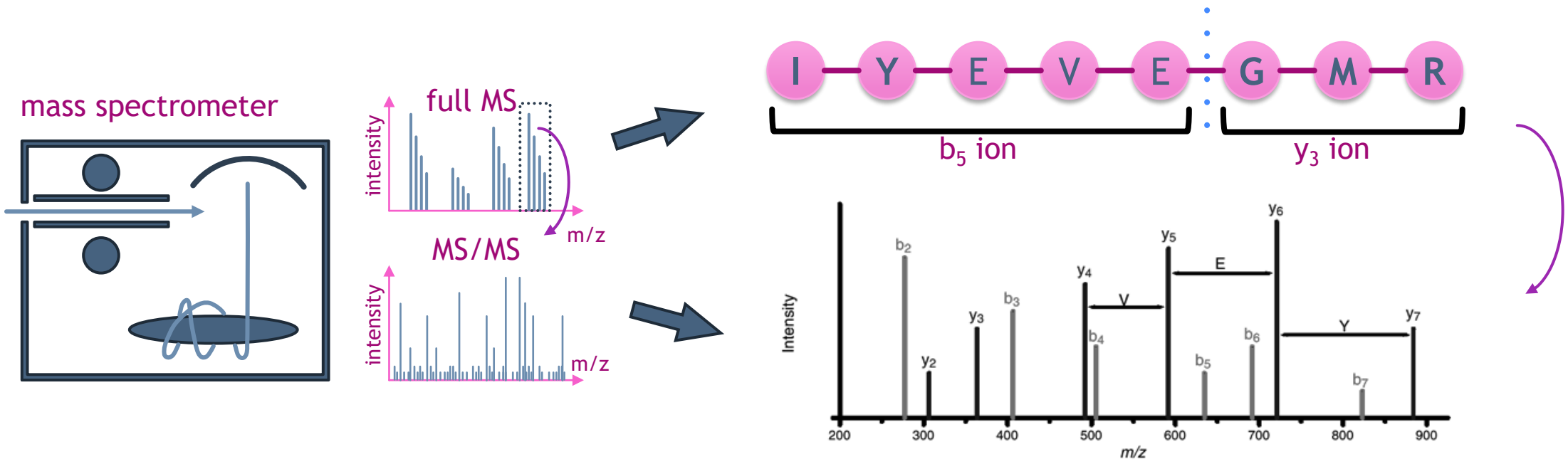


Proteomics Workflow



Pepide Identification by Tandem MS

- Sequence consists of the **same 20 building blocks** (amino acids)
- CID: peptide breaks preferentially along the **backbone**
- Peptide **fragment ions correspond to prefixes and suffixes** of the whole peptide sequences
- Complete ion series (ladders) reveal the sequence via mass differences of adjacent fragment ions



Simulation of MS2 Fragmentation

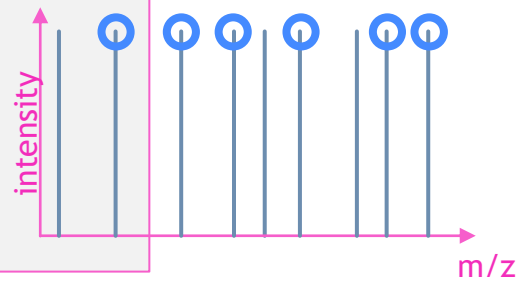
preselecting peptides of interest

F#

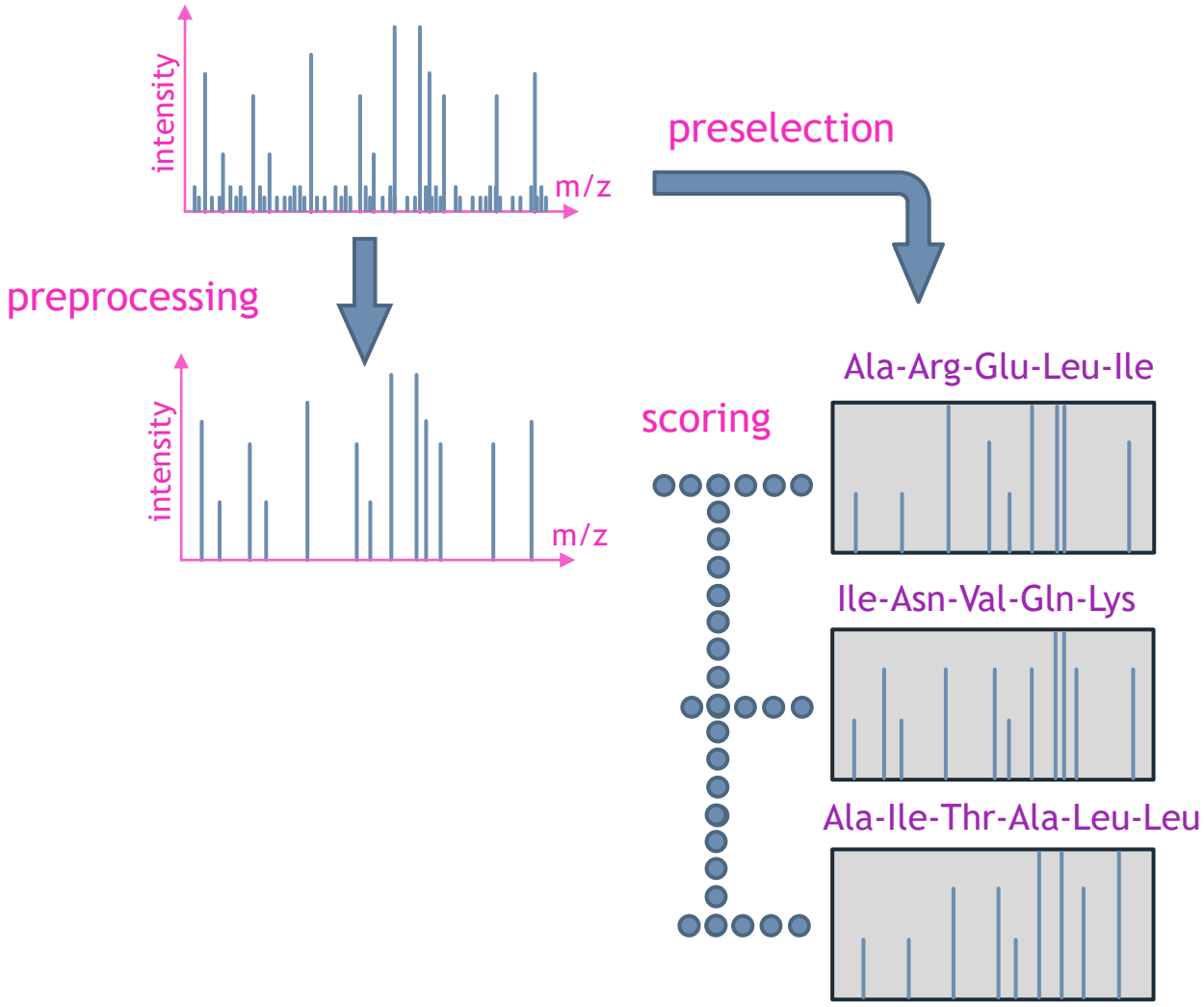
```
peptideAndMasses
|> Array.filter (fun (sequence,mass) -> mass > 1020.52  && mass < 1020.53)
```

```
let predictFromSequence peptide =
    [
        peptide
        |> Mz.Fragmentation.Series.yOfBioList BioItem.initMonoisoMassWithMemP
        peptide
        |> Mz.Fragmentation.Series.bOfBioList BioItem.initMonoisoMassWithMemP
    ]
    |> List.concat
    |> Mz.SequestLike.predictOf (lowerScanLimit,upperScanLimit) 2.
```

hypothetical spectrum
(y/b ions)



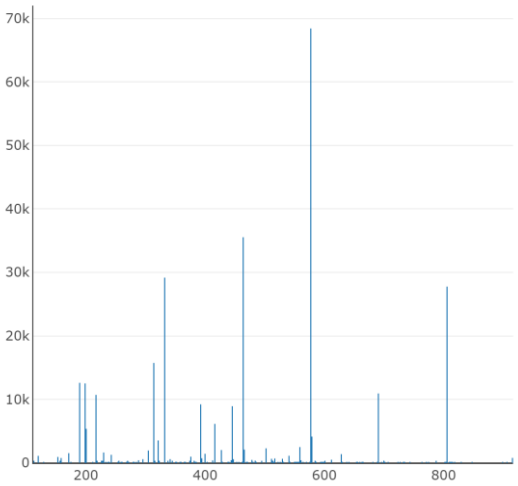
Basic Steps of Peptide Identification

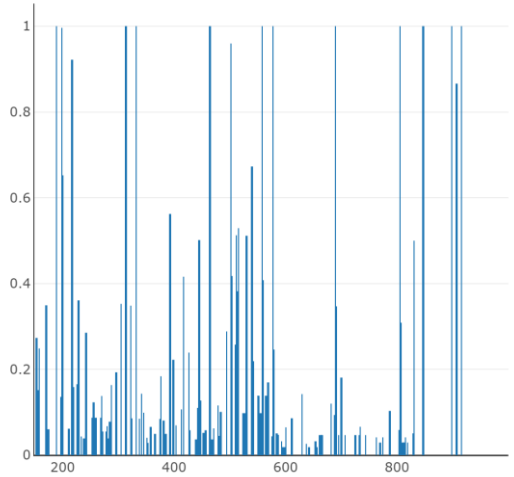


Spectrum Preprocessing

```
let ms2 = BioFSharp.IO.Mgf.readMgf „filename“
```

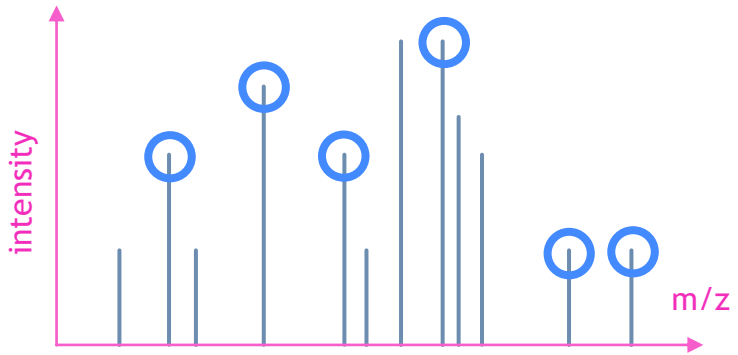
```
let lowerScanLimit = 150.
let upperScanLimit = 1000.

let preprocessedIntesities =
    Mz.PeakArray.zip ms2.Mass ms2.Intensity
    |> (fun pa -> Mz.PeakArray.peaksToNearestUnitDaltonBinVector
                            pa lowerScanLimit upperScanLimit)
    |> (fun pa -> Mz.SequestLike.windowNormalizeIntensities pa 10)

Chart.Column(preprocessedIntesities, [lowerScanLimit .. upperScanLimit])
|> Chart.withTemplate ChartTemplates.light
```



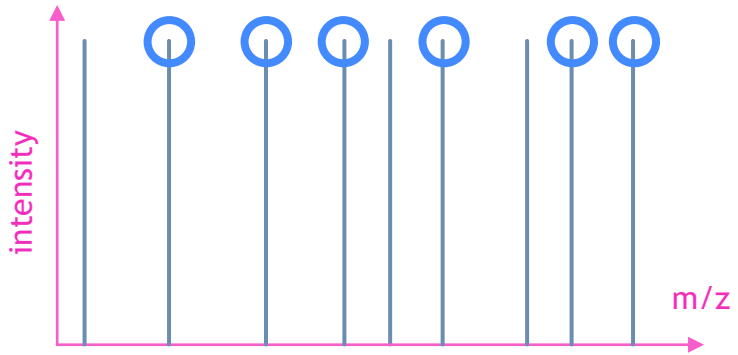


Scoring by Auto-Correlation (SEQUEST)

measured spectrum



acquired spectrum



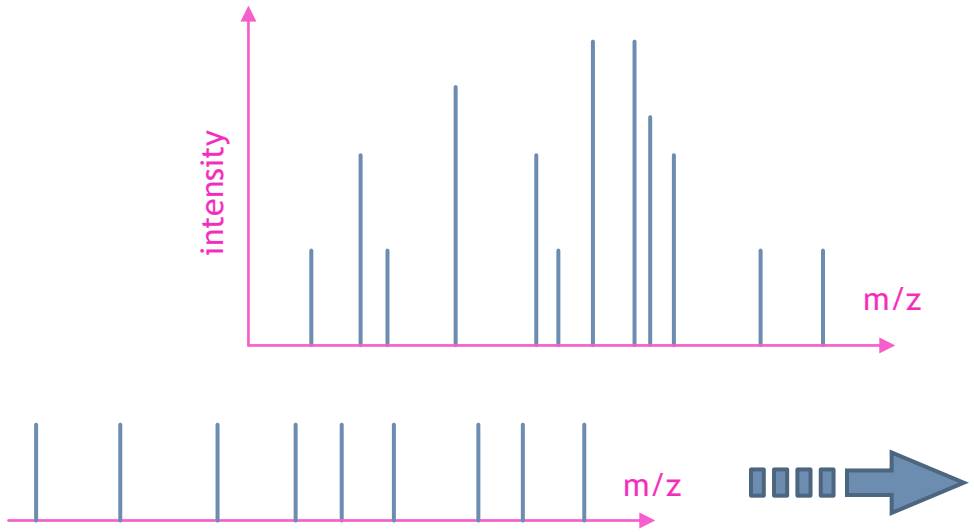
Simulated spectrum

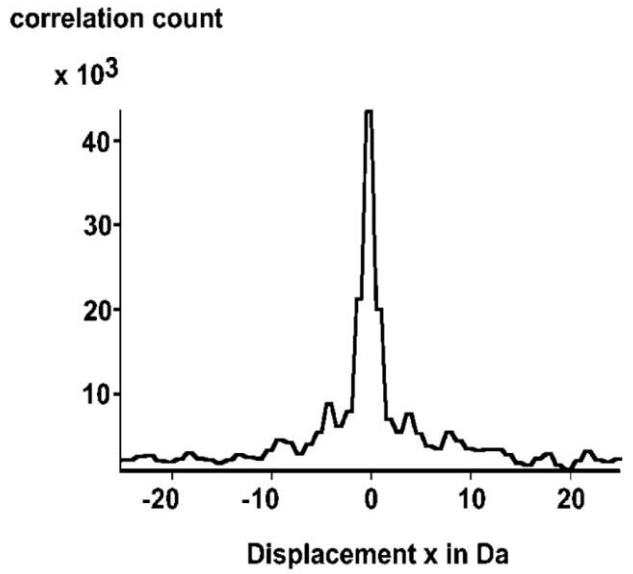
hypothetical spectrum (y/b ions)

cross correlation

sum of all peaks in overlap

Scoring by Auto-Correlation (SEQUEST)





- The peaks that overlap upon spectra shifting are used to calculate the autocorrelation

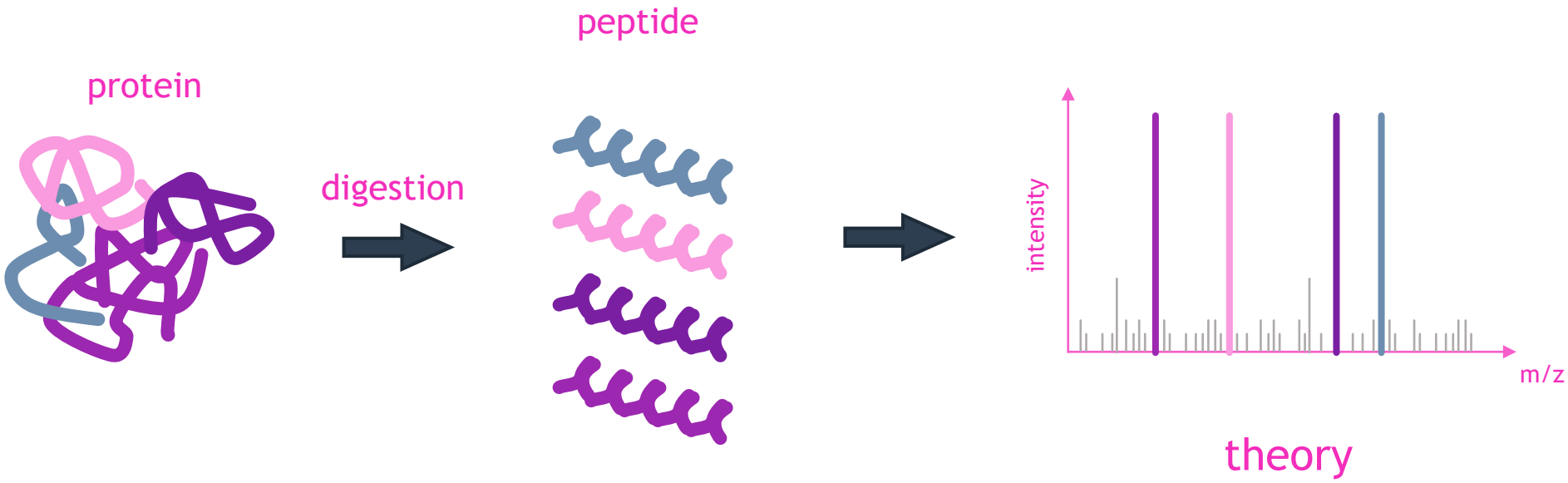
Matching and Scoring

F#

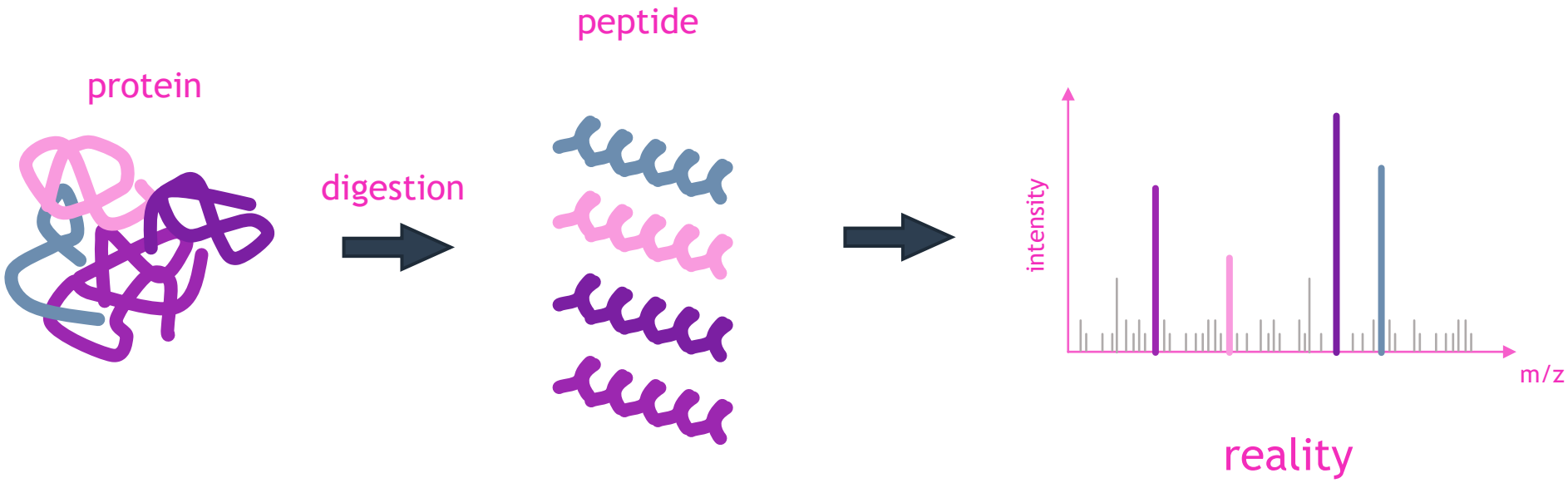
```
let sortedScores =
    peptideAndMasses
    |> Array.filter (fun (sequence,mass) ->
        mass > 1020.52  && mass < 1020.53
    )
    |> Array.map (fun (sequence,mass)    ->
        sequence,predictFromSequence sequence
    )
    |> Array.map (fun (sequence,theoSpectrum) ->
        sequence, BioFSharp.Mz.SequestLike.scoreSingle
                        theoSpectrum preprocessedIntesities
    )
    |> Array.sortByDescending (fun (sequence,score) -> score)

sortedScores
```

index	value
0	▶ *([Asp; Thr; Asp; ...], 11.489317931131332)*
1	▶ *([Val; Val; Asp; ...], 4.800498735800271)*
2	▶ *([Asn; Tyr; Val; ...], 3.547344985238297)*
3	▶ *([Ser; Leu; Ile; ...], 3.0626381054538503)*
4	▶ *([Gly; Glu; Glu; ...], 2.6674782152072827)*
5	▶ *([Asp; Phe; Val; ...], 2.5897868421493198)*
6	▶ *([Leu; Ser; Ser; ...], 2.5532097910662954)*
7	▶ *([Leu; Ser; Ser; ...], 2.5532097910662954)*
8	▶ *([Ala; Val; Gln; ...], 2.5501482345520454)*
9	▶ *([Glu; Tyr; Gln; ...], 2.222608555919107)*
10	▶ *([Thr; Thr; Pro; ...], 1.9836285155579028)*
11	▶ *([Leu; Glu; Gly; ...], 1.9032621441376565)*
12	▶ *([Asp; Val; Leu; ...], 1.7938301080315675)*
13	▶ *([Phe; Leu; Asp; ...], 1.7615553265303323)*
14	▶ *([Val; Asp; Phe; ...], 1.7576420105210504)*
15	▶ *([Ala; Glu; Ala; ...], 1.6090022149128864)*
16	▶ *([Leu; Gly; Ala; ...], 1.5177524771864663)*
17	▶ *([Gln; Ile; Ile; ...], 1.5004855067669207)*
18	▶ *([Trp; Pro; Gly; ...], 1.4586761816897038)*
19	▶ *([Val; Ser; Thr; ...], 1.1391486393077925)*

Detectability Problem for Peak Intensities



Detectability Problem for Peak Intensities

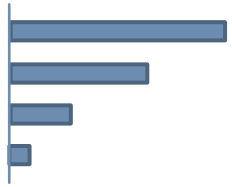


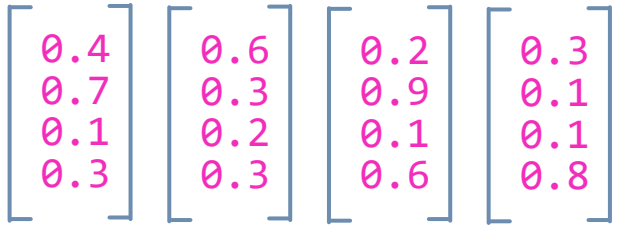
d::pPop algorithm

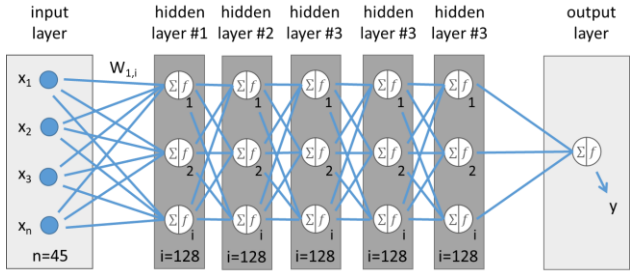
protein A

LGREWELSFRK
SGTFNFGDK
VALTALTAK
NKQDVLFFIDNIK

intensity





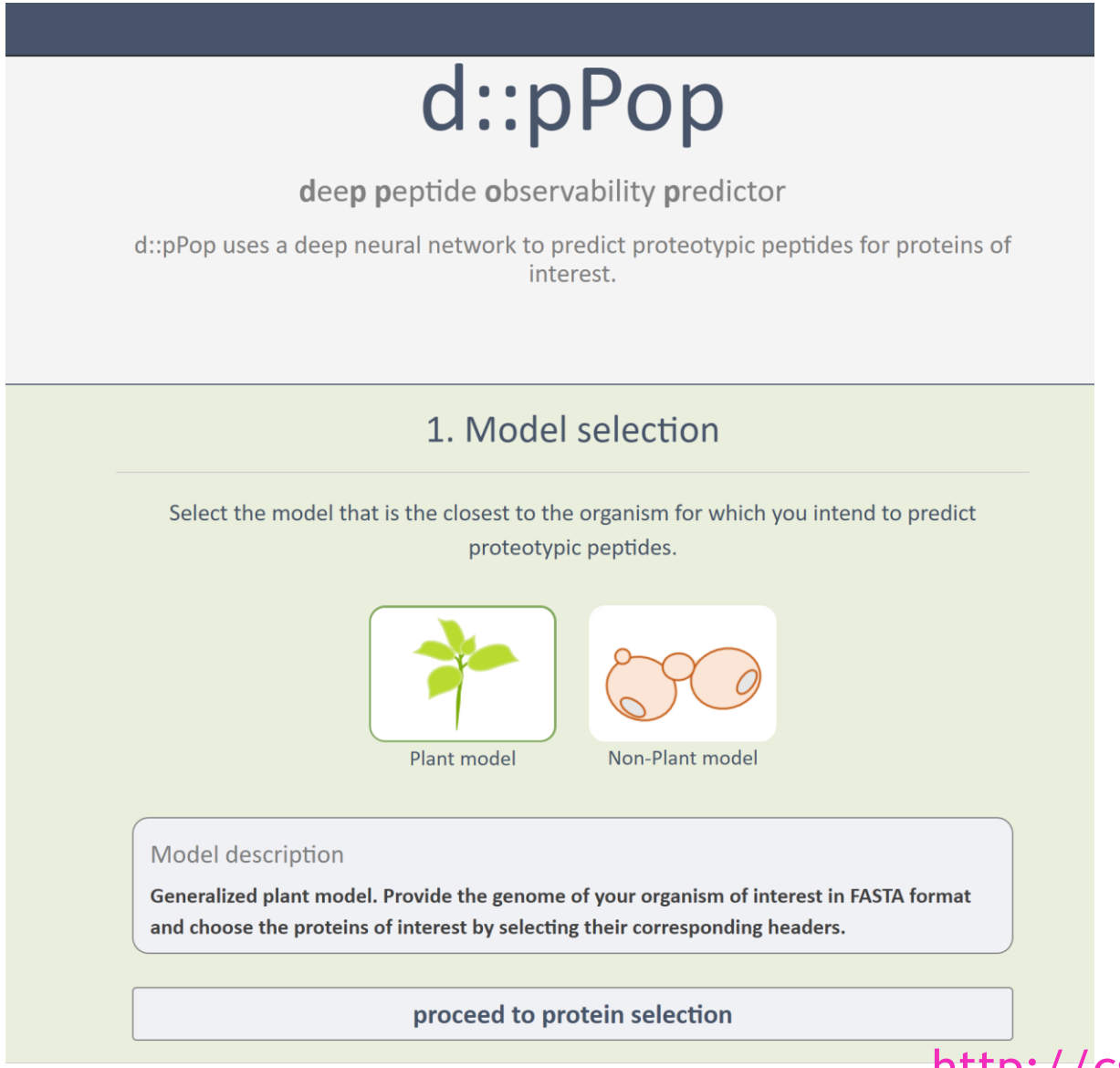


* ranking

* feature vectors

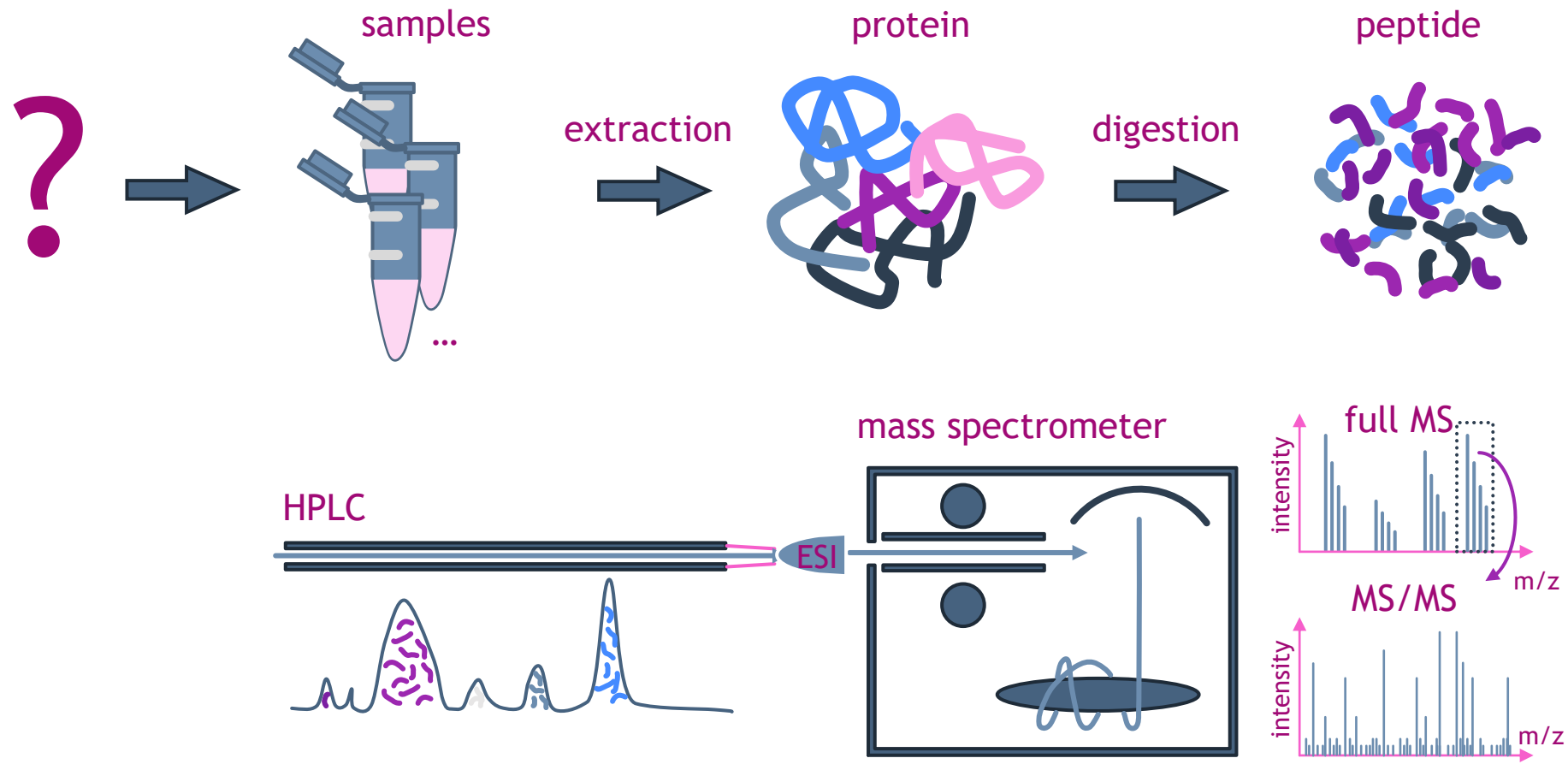
* machine learning

d::pPop website

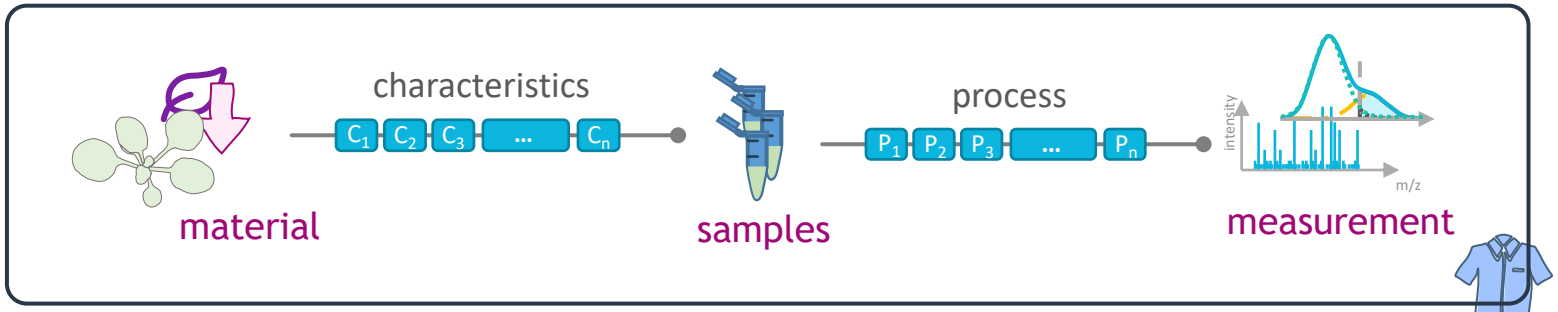


http://csbweb.bio.uni-kl.de

Proteomics Workflow



Annotation of the Sample Generating Process



administrative metadata

Sample	Parameter [instrument model]	Data
	´TripleTOF4600 ’	
	´TripleTOF4600 ’	
	´TripleTOF4600 ’	
	´TripleTOF4600 ’	
	´TripleTOF4600 ’	

A B C D

Swate – Metadata annotation tool

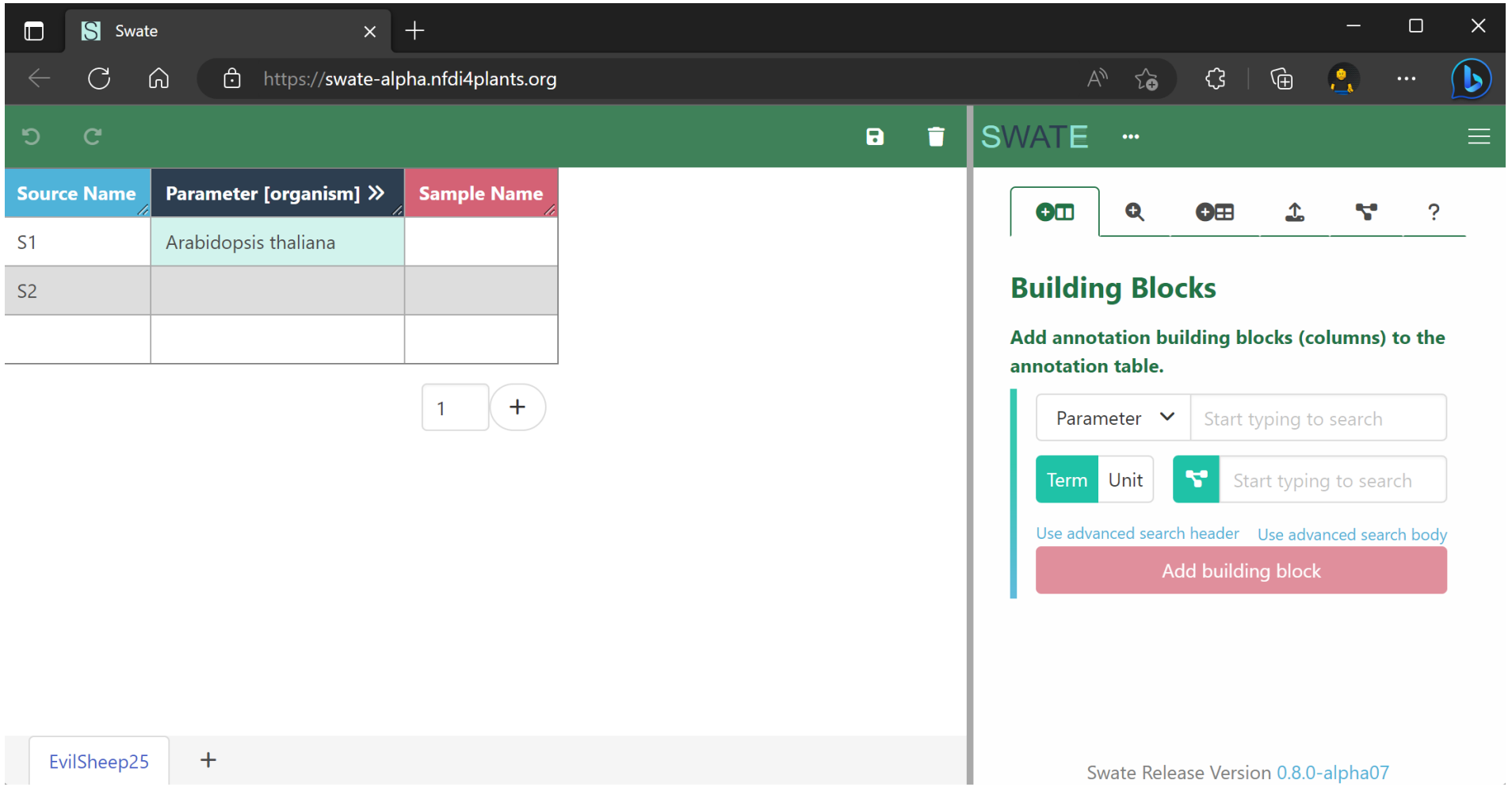
Sink | Parameter [´parent term’] | Source

A

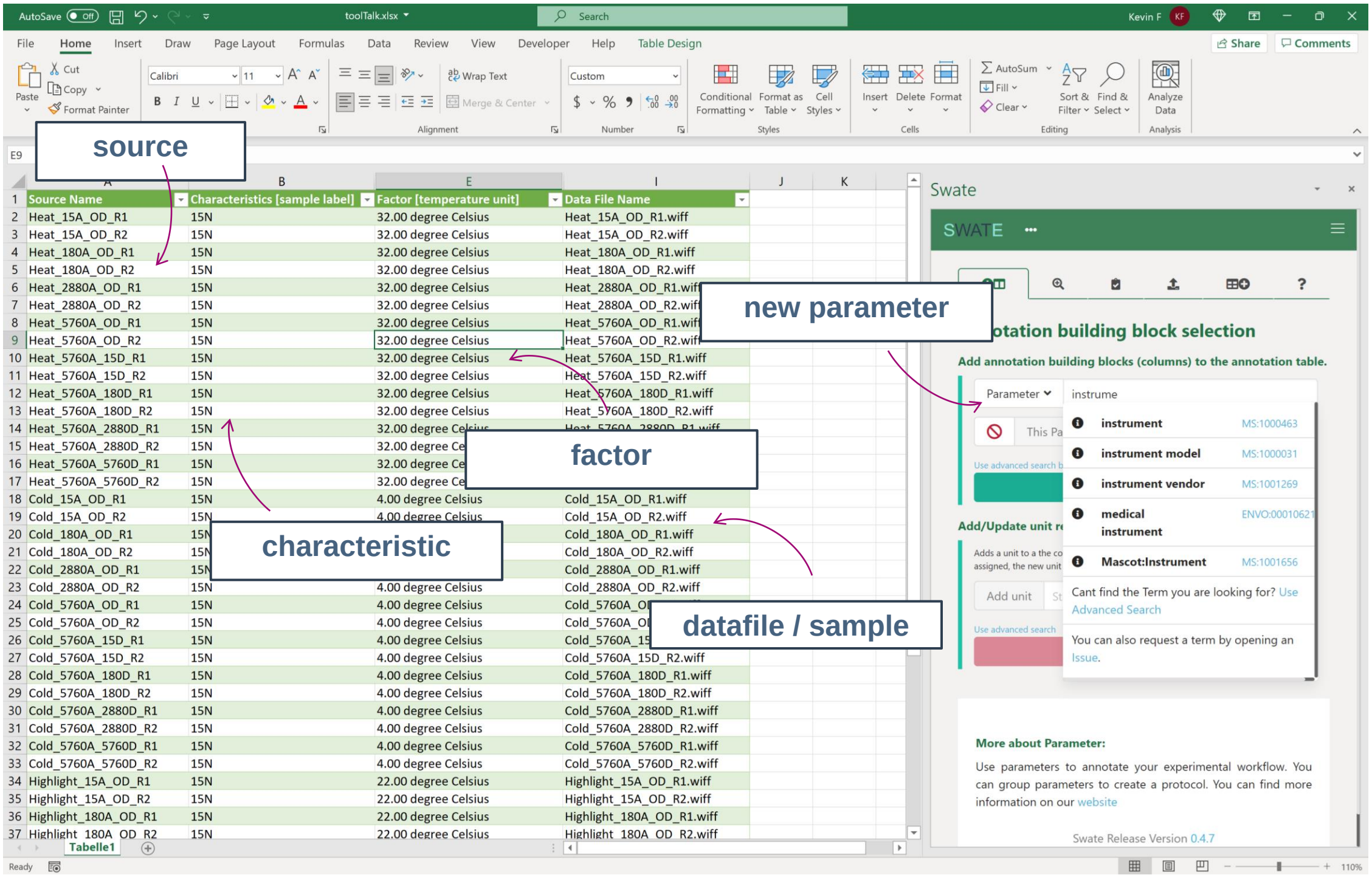
new parameter

Excel + Swate Plug-In

Swate without Excel

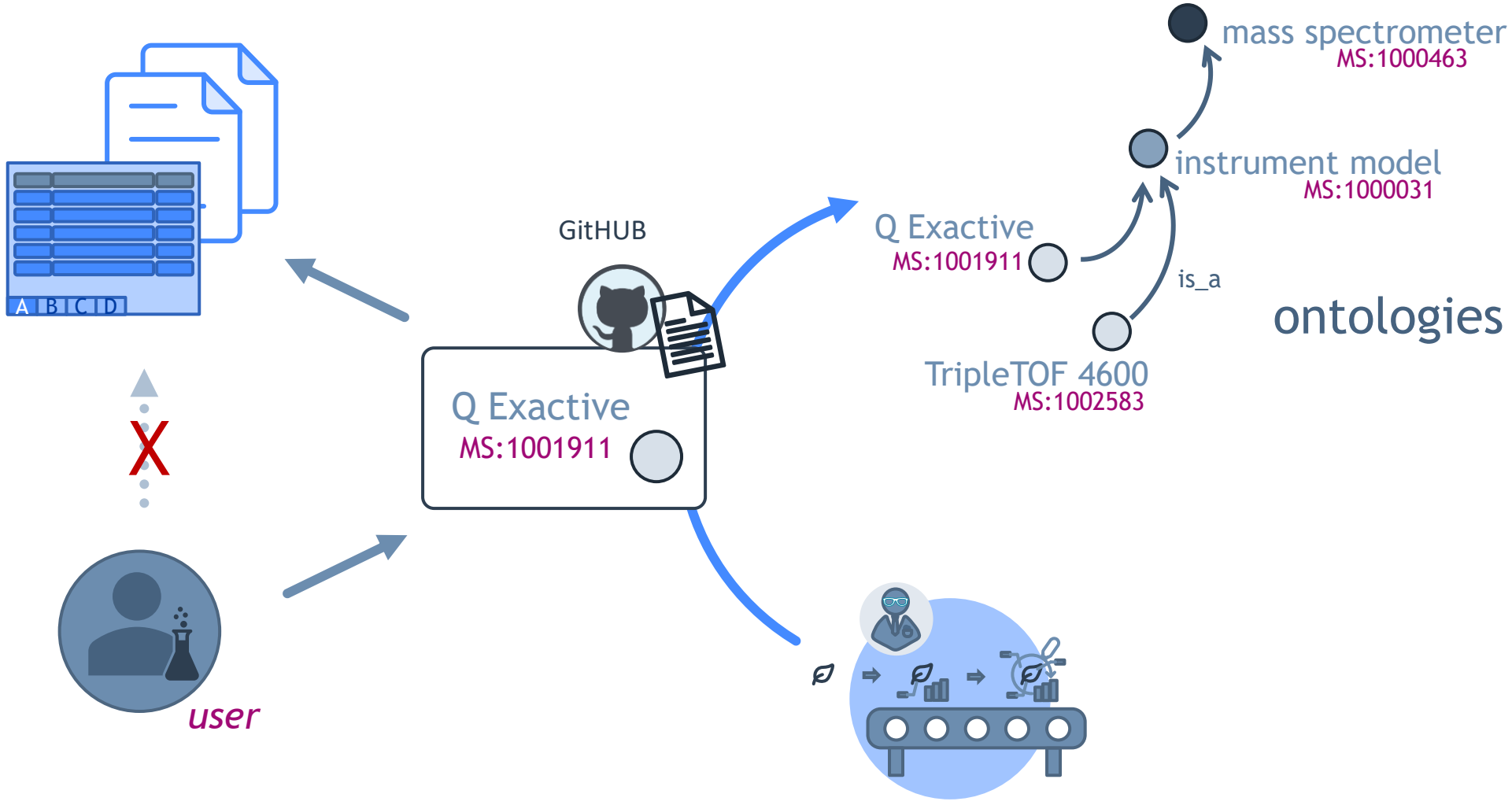


alpha.nfdi4plants.org



Source Name	Characteristics [sample label]	Factor [temperature unit]	Data File Name
Heat_15A_OD_R1	15N	32.00 degree Celsius	Heat_15A_OD_R1.wiff
Heat_15A_OD_R2	15N	32.00 degree Celsius	Heat_15A_OD_R2.wiff
Heat_180A_OD_R1	15N	32.00 degree Celsius	Heat_180A_OD_R1.wiff
Heat_180A_OD_R2	15N	32.00 degree Celsius	Heat_180A_OD_R2.wiff
Heat_2880A_OD_R1	15N	32.00 degree Celsius	Heat_2880A_OD_R1.wiff
Heat_2880A_OD_R2	15N	32.00 degree Celsius	Heat_2880A_OD_R2.wiff
Heat_5760A_OD_R1	15N	32.00 degree Celsius	Heat_5760A_OD_R1.wiff
Heat_5760A_OD_R2	15N	32.00 degree Celsius	Heat_5760A_OD_R2.wiff
Heat_5760A_15D_R1	15N	32.00 degree Celsius	Heat_5760A_15D_R1.wiff
Heat_5760A_15D_R2	15N	32.00 degree Celsius	Heat_5760A_15D_R2.wiff
Heat_5760A_180D_R1	15N	32.00 degree Celsius	Heat_5760A_180D_R1.wiff
Heat_5760A_180D_R2	15N	32.00 degree Celsius	Heat_5760A_180D_R2.wiff
Heat_5760A_2880D_R1	15N	32.00 degree Celsius	Heat_5760A_2880D_R1.wiff
Heat_5760A_2880D_R2	15N	32.00 degree Celsius	
Heat_5760A_5760D_R1	15N	32.00 degree Celsius	
Heat_5760A_5760D_R2	15N	32.00 degree Celsius	
Cold_15A_OD_R1	15N	4.00 degree Celsius	Cold_15A_OD_R1.wiff
Cold_15A_OD_R2	15N	4.00 degree Celsius	Cold_15A_OD_R2.wiff
Cold_180A_OD_R1	15N		Cold_180A_OD_R1.wiff
Cold_180A_OD_R2	15N		Cold_180A_OD_R2.wiff
Cold_2880A_OD_R1	15N		Cold_2880A_OD_R1.wiff
Cold_2880A_OD_R2	15N	4.00 degree Celsius	Cold_2880A_OD_R2.wiff
Cold_5760A_OD_R1	15N	4.00 degree Celsius	Cold_5760A_OD
Cold_5760A_OD_R2	15N	4.00 degree Celsius	Cold_5760A_OD
Cold_5760A_15D_R1	15N	4.00 degree Celsius	Cold_5760A_15
Cold_5760A_15D_R2	15N	4.00 degree Celsius	Cold_5760A_15D_R2.wiff
Cold_5760A_180D_R1	15N	4.00 degree Celsius	Cold_5760A_180D_R1.wiff
Cold_5760A_180D_R2	15N	4.00 degree Celsius	Cold_5760A_180D_R2.wiff
Cold_5760A_2880D_R1	15N	4.00 degree Celsius	Cold_5760A_2880D_R1.wiff
Cold_5760A_2880D_R2	15N	4.00 degree Celsius	Cold_5760A_2880D_R2.wiff
Cold_5760A_5760D_R1	15N	4.00 degree Celsius	Cold_5760A_5760D_R1.wiff
Cold_5760A_5760D_R2	15N	4.00 degree Celsius	Cold_5760A_5760D_R2.wiff
Highlight_15A_OD_R1	15N	22.00 degree Celsius	Highlight_15A_OD_R1.wiff
Highlight_15A_OD_R2	15N	22.00 degree Celsius	Highlight_15A_OD_R2.wiff
Highlight_180A_OD_R1	15N	22.00 degree Celsius	Highlight_180A_OD_R1.wiff
Highlight_180A_OD_R2	15N	22.00 degree Celsius	Highlight_180A_OD_R2.wiff

Crowdsourcing controlled vocabulary development



Computational Systems Biology

TRR 175

SPONSORED BY THE
Federal Ministry of Education and Research

nfdi Nationale Forschungsdaten Infrastruktur

bioComp

Deutsche Forschungsgemeinschaft DFG

Thank you for your attention