```
using DrWatson
```

```
@quickactivate "SensitivityEstimate"
```

▸ Dict(:isotopeName ⇒ "2nubb", :activity ⇒ $0.001721 \pm 3.1e-5$, :nTotalSim ⇒ 1.0e8, :timeM

```
begin
    using StatsPlots, UnROOT, DataFramesMeta, LaTeXStrings, Revise, StatsBase, FHist
    , Measurements
    push!(LOAD_PATH, srcdir())
    using SensitivityModule
    include(scriptsdir("Params.jl"))
end
```

```
begin
    _palette = ["#00a0f9", "#ba3030", "#22ac74", "#707070", "#9452bd", "#d76055"]
    theme(
        :dao;
        size = (1200, 800),
        legend = :best,
        guidefontsize = 16,
        tickfontsize = 12,
        titlefontsize = 16,
        legendfontsize = 12,
        left_margin = 4Plots.mm,
        right_margin = 8Plots.mm,
        top_margin = 4Plots.mm,
        bottom_margin = 6Plots.mm,
        thickness_scaling = 1.4,
        :linewidth => 3,
        dpi = 200,
        widen = :true,
        :markerstrokewidth => 1,
        :markerstrokecolor => :black,
        :palette => _palette
    )
end
```

First we load all the `.root` input simulation files:

1. **Background processes** (all generated on source foil):

- Bi214
- Tl208
- K40
- Pa234m

2. **Signal process**:

refined spectrum ξ_{31}, ξ_{51} with the following parameters:

- $\xi_{31} = 0.37$
- $\xi_{51} = 0.14$
- $K^{2\nu} = -0.66$

```
ROOTFile with 1 entry and 20 streamers.
/home/shoram/Work/PhD_Thesis/SNSensitivityEstimate/data/sims/Xi037_EnePhiDistPos_J23.root
└─ tree (TTree)
   ├─ "phi"
   ├─ "reconstructedEnergy1"
   ├─ "reconstructedEnergy2"
   ├─ "⋮"
   ├─ "x2Escaped"
   ├─ "y2Escaped"
   └─ "z2Escaped"
```

```
begin
    Bi214file = ROOTFile(datadir("sims", "Bi214_EnePhiDistPos_J23.root"))
    Tl208file = ROOTFile(datadir("sims", "Tl208_EnePhiDistPos_J23.root"))
    K40file   = ROOTFile(datadir("sims", "K40_EnePhiDistPos_J23.root"))
    Pa234mfile= ROOTFile(datadir("sims", "Pa234m_EnePhiDistPos_J23.root"))
    bbfile    = ROOTFile(datadir("sims", "Xi037_EnePhiDistPos_J23.root"))
end
```

Now we transform the raw inptu data into `DataFrame` format for easier handling.

Here we also import only three variables: E_1, E_2 and ϕ

	reconstructedEnergy2	reconstructedEnergy1	phi
1	978.629	141.823	91.9659
2	353.124	897.696	138.051
3	772.878	1080.25	129.36
4	614.549	1075.65	69.5691
5	714.981	172.517	22.4111
6	497.665	456.098	15.5818
7	352.436	465.499	99.5599
8	418.158	608.005	91.5183
9	874.786	263.201	72.1372
10	925.715	405.297	146.289
⋮	more		
8240117	503.707	504.553	157.208

```
begin
    Bi214 = fill_from_root_file(
        Bi214file, "tree", ["phi", "reconstructedEnergy1", "reconstructedEnergy2"]
    )

    Tl208 = fill_from_root_file(
        Tl208file, "tree", ["phi", "reconstructedEnergy1", "reconstructedEnergy2"]
    )

    K40 = fill_from_root_file(
        K40file, "tree", ["phi", "reconstructedEnergy1", "reconstructedEnergy2"]
    )

    Pa234m = fill_from_root_file(
        Pa234mfile, "tree", ["phi", "reconstructedEnergy1", "reconstructedEnergy2"]
    )

    bb = fill_from_root_file(
        bbfile, "tree", ["phi", "reconstructedEnergy1", "reconstructedEnergy2"]
    )
end
```

Next we import all the important parameters for the analysis.

The import itself has already been done in the 3rd cell by the `include(scriptsdit("Params.jl"))` command. Now we analyse what's inside.

The file `Params.jl` contains all the input constants that will be used in the analysis. These are divided into groups:

- **Detector Parameters** (Dict variable `SNParams`):

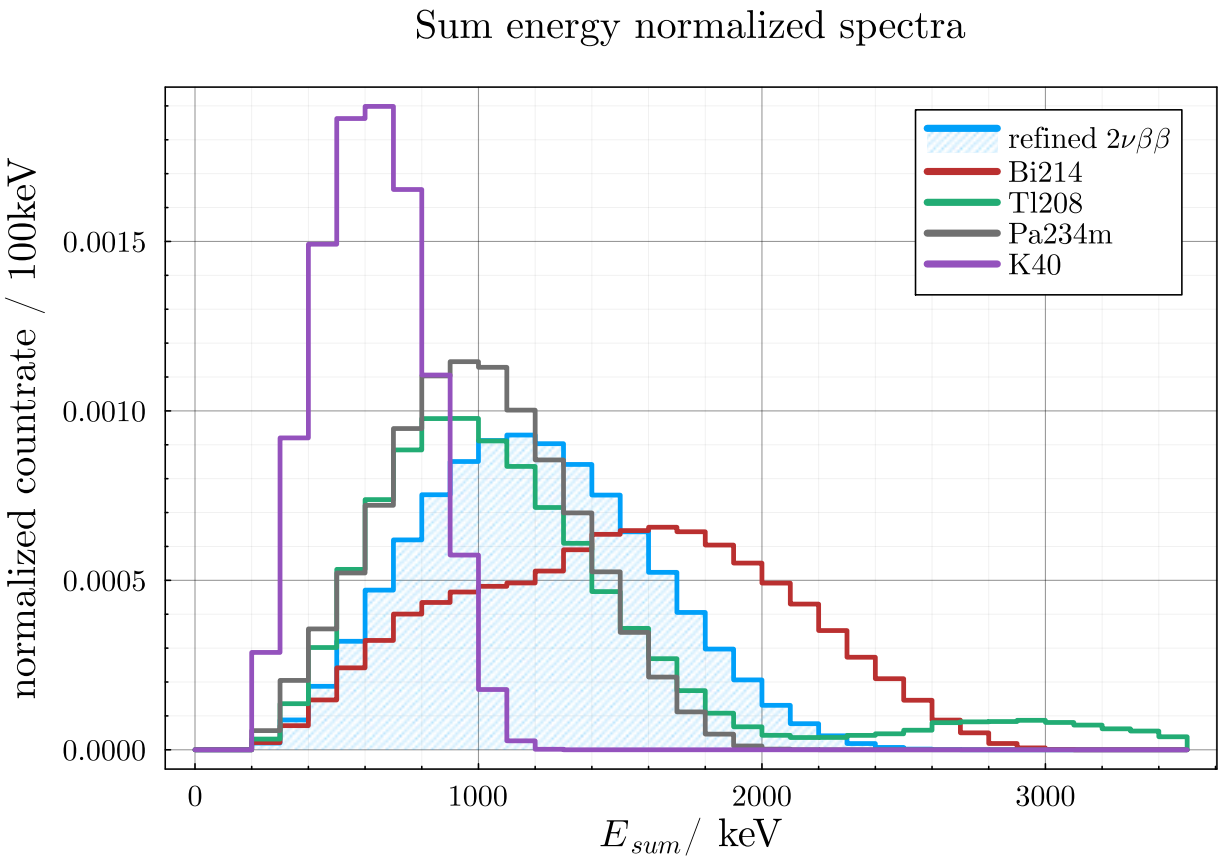
    - N_a = 6.02214e23 → Avogadro's number in [1/mol]
    - W = 0.08192 → Se82 molar mass in [kg/mol]
    - a = 0.99 → abundance/foil enrichment; check number
    - m = 6.25 → foil mass in [kg]
    - t = 7.884e7 → measurement time in [s]
    - tYear = 2.5 → measurement time in [y]
    - SeThalf = 2.961e27 ± 5.4e25 → Se82 half life **in [s]**, results from NEMO-3
- **Activity Parameters** (Dict variables `BkgActivityParams` and `SigActivityParams`):
    - **Background** (Taken from NEMO-3 paper)
        - :Bi214 => 0.0015 ± 4.0e-5 Bq/kg
        - :Tl208 => 0.00039 ± 1.0e-5 Bq/kg
        - :Pa234m => 0.0173 ± 0.0001 Bq/kg
        - :K40 => 0.0587 ± 0.0001 Bq/kg
    - **Signal**
        - :Xi037 => 0.0015 ± 0.0001 Bq/kg # a mock value is used here, the activity in reality is the free parameter of this analysis
- **Histogram Parameters** which depend on the analysis channel:
    - Sum energy => 0:100:3500
    - Single energy => 0:100:3500
    - Phi => 0:5:180

Lastly, for obtaining efficiencies, the histograms are divided by the `nTotalSim` parameter representing the total number of simulated events. In this example, for Sum energy and phi channels $nTotalSim$ = 1.0e8 and for the single energy channel $nTotalSim$ = 2.0e8.

Before starting the actual analysis, let's look at the raw data:

(for now we look only at the sum energy channel)

Below, the **normalized Sum Energy spectra** for each process are depicted. It can be seen that there is quite a bit of overlap over the whole range. However, the picture changes (slightly) if we scale the spectra according to their activities and measurement time, thus obtaining an **estimate** of the real spectrum, which SuperNEMO will measure. (Considering only the depicted processes...)



Now the estimated total spectrum as would be seen by SuperNEMO:

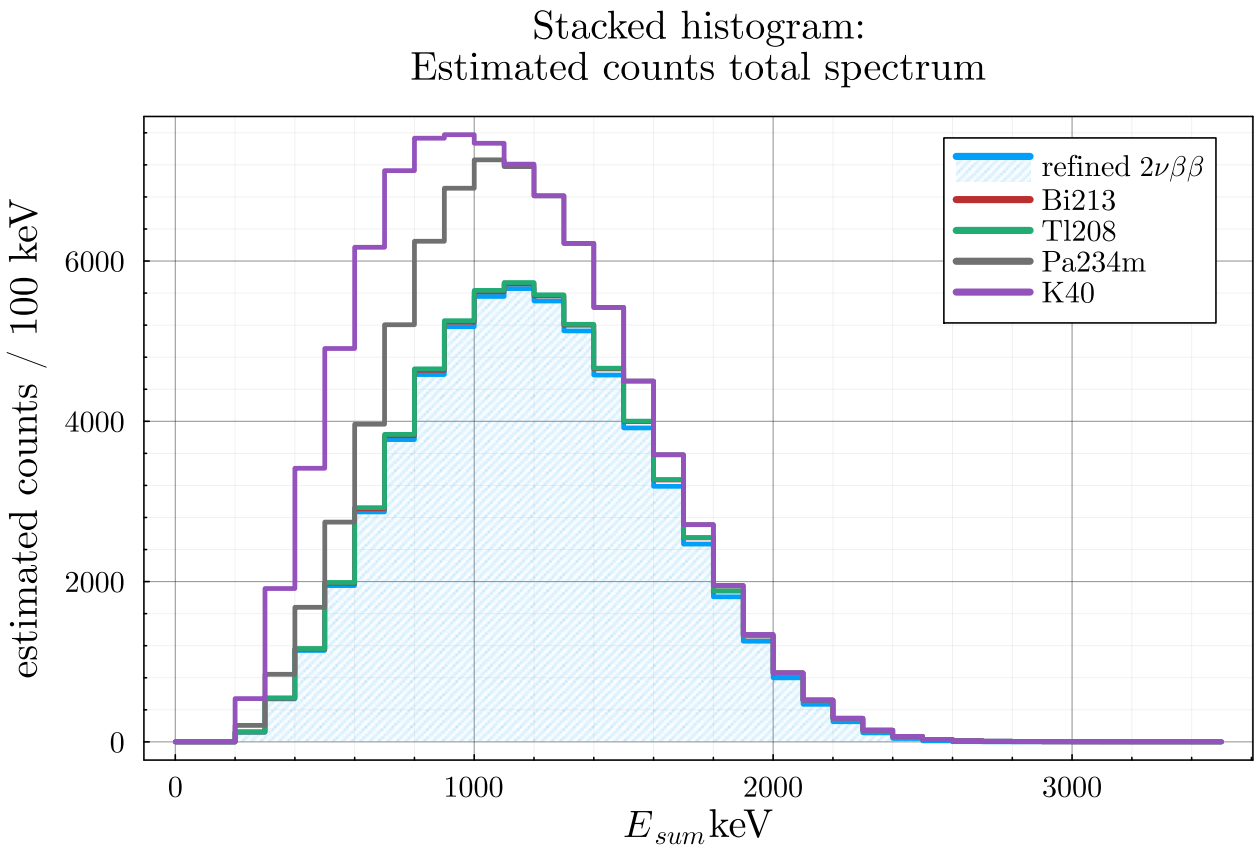
The estimated **total** counts for the full spectrum for each process can be calculated as:

$$n = A * t * m * \varepsilon$$

With the previously summarized parameters this is:

- Bi214: n = 1204.0
- Tl208: n = 191.0
- Pa234m: n = 14434.0
- K40: n = 11629.0
- Xi31: n = 60905.0

Notice that the main contributions to the background for $\mathbf{2\nu\beta\beta}$ come from K40 and Pa234m processes mainly in the lower energy region.



Analysis

To perform the analysis as described in `docdb:5833`. We must follow the steps:

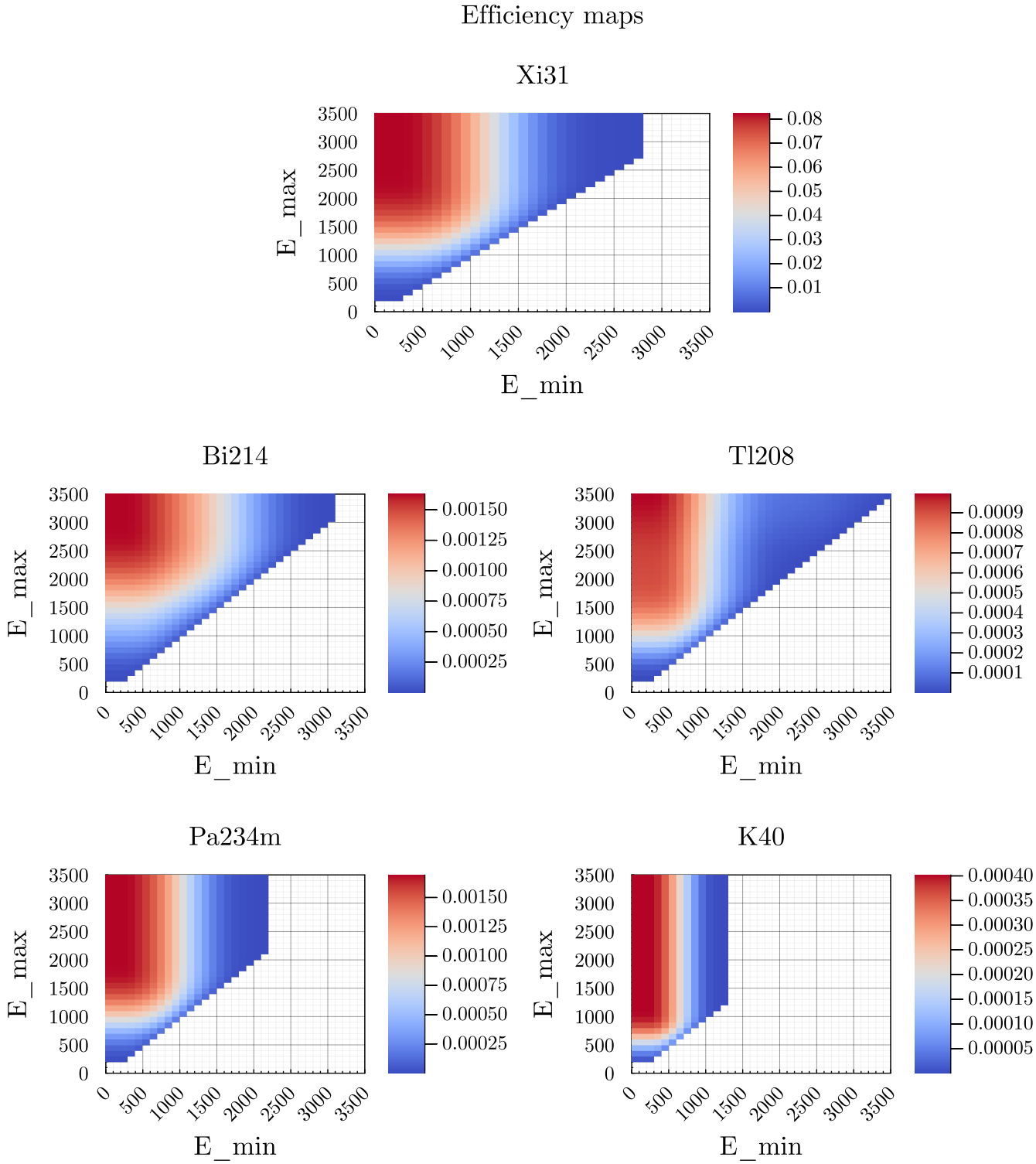
1. calculate the efficiency maps
2. calculate the corresponding s/b ratios
3. determine best ROI
4. extract $\bar{b}$ and ε in the ROI
5. calculate $T_{1/2}$

In this analysis, I've created a data-type `Process` which holds the information about the individual studied processes and I've written a number of methods which calculate the efficiency maps, ROIs and T12. The following example shows the pipeline to obtain the T12:

First we instantiate the `Process`'s for each isotope:

```
begin
    Bi214SumE = Process(
        Bi214.reconstructedEnergy1 .+ Bi214.reconstructedEnergy2, sumEBi214Params
    )

    Tl208SumE = Process(
        Tl208.reconstructedEnergy1 .+ Tl208.reconstructedEnergy2, sumETl208Params
    )

    K40SumE = Process(
        K40.reconstructedEnergy1 .+ K40.reconstructedEnergy2, sumEK40Params
    )

    Pa234mSumE = Process(
        Pa234m.reconstructedEnergy1 .+ Pa234m.reconstructedEnergy2, sumEPa234mParams
    )

    bbSumE = Process(
        bb.reconstructedEnergy1 .+ bb.reconstructedEnergy2, sumEbbParams
    )
    nothing # so there's no cell output
end
```

At construction, the efficiency maps are created as well:



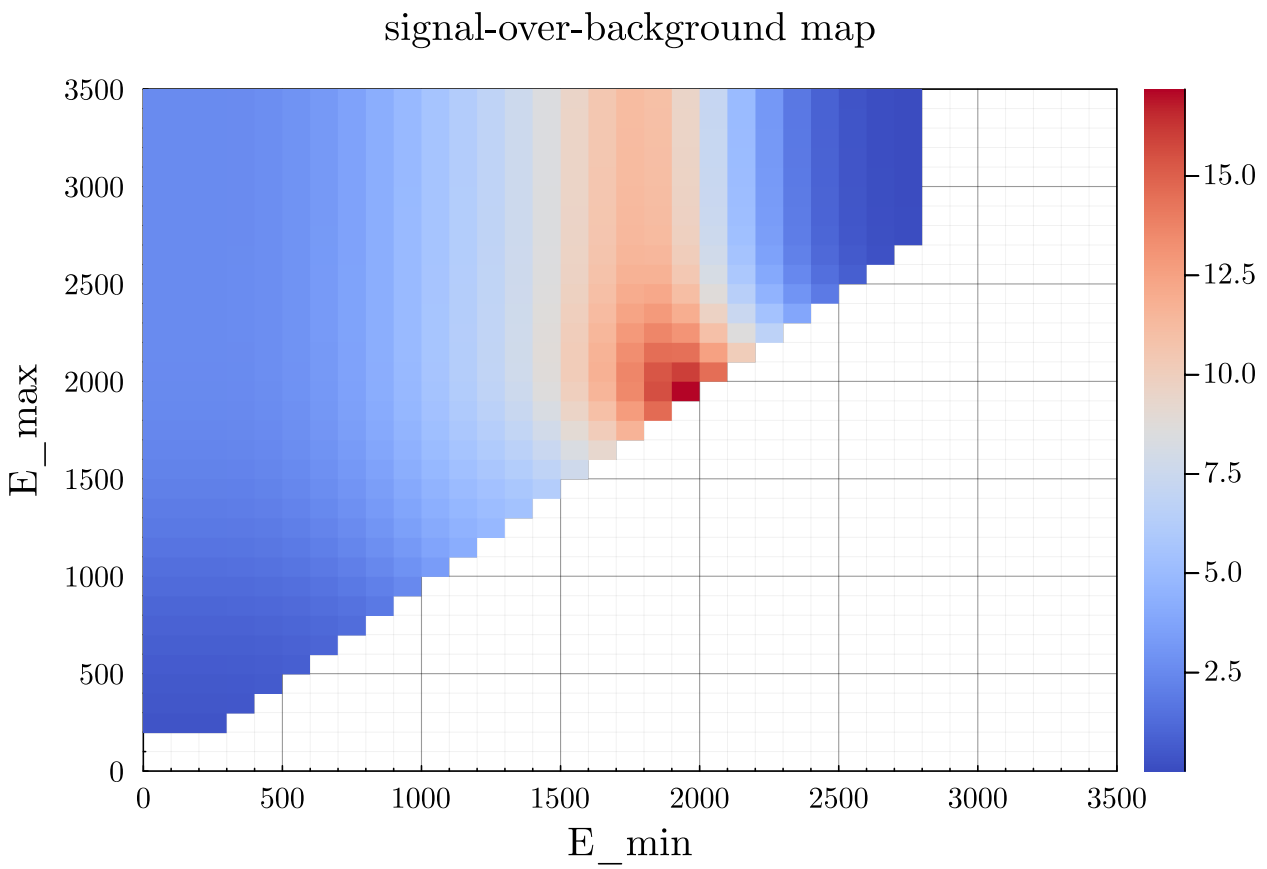

```
let
    p1 = plot(bbSumE.efficiency, widen =:false, c =:coolwarm, xrotation=45, title=
    "Xi31")
    p2 = plot(Bi214SumE.efficiency, widen =:false, c =:coolwarm, xrotation=45,
    title= "Bi214")
    p3 = plot(Tl208SumE.efficiency, widen =:false, c =:coolwarm, xrotation=45,
    title= "Tl208")
    p4 = plot(Pa234mSumE.efficiency, widen =:false, c =:coolwarm, xrotation=45,
    title= "Pa234m")
    p5 = plot(K40SumE.efficiency, widen =:false, c =:coolwarm, xrotation=45, title=
    "K40")
```

```
    l = @layout [_ a{0.6w} _
                 b c
                 d e]

    plot(p1, p2, p3, p4, p5,
        size = (1400, 1600),layout = l,
        xlabel ="E_min", ylabel ="E_max",
        plot_title = "Efficiency maps",
        bottom_margins = 1Plots.px,
        top_margins = 1Plots.px,
        left_margins = 1Plots.px,
        right_margins = 20Plots.px,
    )
end
```

Combining the **estimated** counts into the efficiency maps we can create the s/b maps:

```
stbSum = get_sToBRatio(bbSumE, Bi214SumE, Tl208SumE, Pa234mSumE, K40SumE);
```

```
⚠ get_bkg_rate(): passed isotope 2nubb is a signal process!!
⚠ get_sig_rate(): passed isotope Bi214 is a background process!!
⚠ get_sig_rate(): passed isotope Tl208 is a background process!!
⚠ get_sig_rate(): passed isotope Pa234m is a background process!!
⚠ get_sig_rate(): passed isotope K40 is a background process!!
```




```
plot(
    stbSum,
    c =:coolwarm, widen =:false,
    xlabel ="E_min", ylabel ="E_max",
    title= "signal-over-background map"
)
```

Largest s/b value (best ROI) is found with:

- ROI: (1900.0 - 2000.0) keV
- s/b: 17.0

```
best_stbSum= get_max_bin(stbSum);
```

The estimated background count in the ROI is: b = 84.0 ± 2.0

```
expBkgESum = get_estimated_bkg_counts(best_stbSum, SNparams, Bi214SumE, Tl208SumE,
Pa234mSumE, K40SumE);

```

The corresponding signal efficiency in ROI is: 0.00278036

```
effbb = lookup( bbSumE.efficiency, best_stbSum[:minBinEdge], best_stbSum[:maxBinEdge]
);
```

Finally the calculated sensitivity (assuming Gaussian approximation) for CL = 90% is:

$$T_{1/2}^{2\nu} \geq (1.33 \pm 0.01) \cdot 10^{22}\text{yr}$$

ThalfbbESum =

$$1.33e22 \pm 1.4e20$$

```
ThalfbbESum = get_tHalf(SNparams, effbb, expBkgESum, 1.8)

```