Predicting automobile insurance claims

```
PGFPlotsXBackend()
```

```
begin
    using DataFrames
    using Empirikos
    using MosekTools
    using JuMP
    using LaTeXStrings
    using Plots
    using PGFPlotsX
    pgfplotsx()
end
```

```
quiet_mosek =
  OptimizerWithAttributes(Optimizer (generic function with 2 methods), [RawParameter("MSK_D
```

```
quiet_mosek = optimizer_with_attributes(Mosek.Optimizer,
                    "MSK_DPAR_INTPNT_CO_TOL_REL_GAP" => 10^(-15))
```

```
Zs_keys = [PoissonSample.(0:4); PoissonSample(Interval(5,nothing))];
```

```
Ns =  [103704, 14075, 1766, 255, 45, 8]
```

```
Ns = [103704; 14075; 1766; 255; 45; 8]
```

```
Zs =
  MultinomialSummary(SortedDict(Z=0  | E=1.0 ⇒ 103704, Z=1  | E=1.0 ⇒ 14075, Z=2  | E=1.0
```

```
Zs = Empirikos.MultinomialSummary(Zs_keys, Ns)
```

```
postmean_targets =
  [PosteriorMean(Z=0  | E=1.0), PosteriorMean(Z=1  | E=1.0), PosteriorMean(Z=2  | E=1.0), P
```

```
postmean_targets = PosteriorMean.(PoissonSample.(0:4))
```

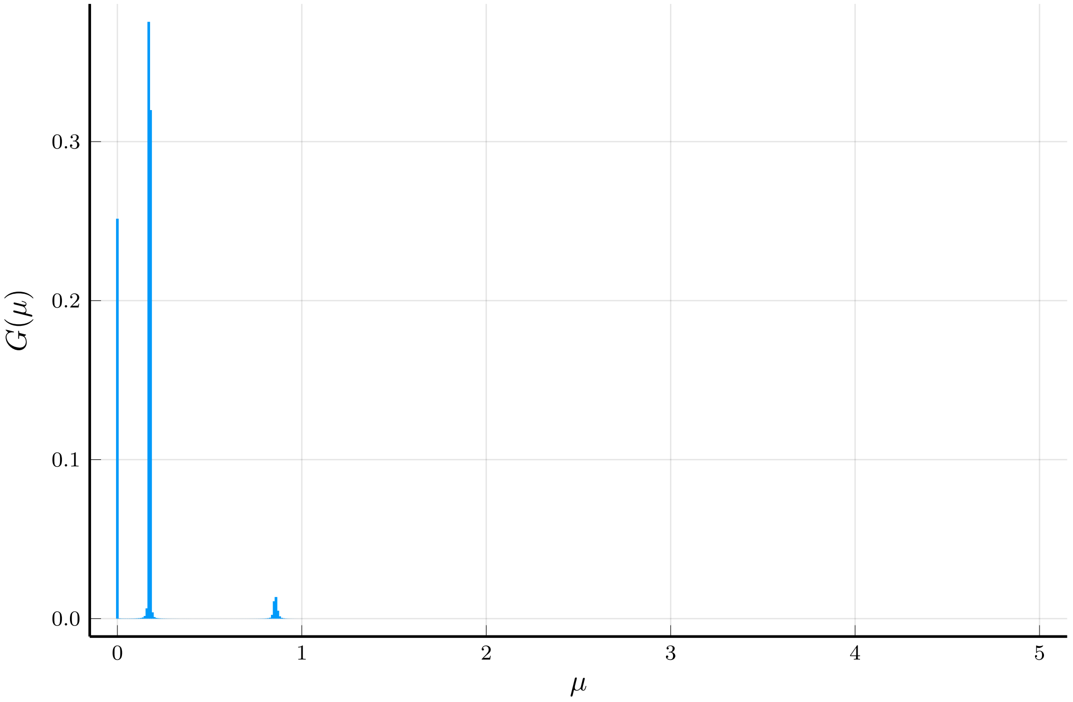
```
𝒢 = DiscretePriorClass | support = 0.0:0.01:5.0
```

```
𝒢 = DiscretePriorClass(0.0:0.01:5.0)
```

Compute NPMLE prior and plug-in posterior mean estimates

```
npmle_fit =
Fitted NPMLE with MathOptInterface.OptimizerWithAttributes(Mosek.Optimizer, Pair{MathOptIn
DiscretePriorClass | support = 0.0:0.01:5.0
```

```
npmle_fit = fit(NPMLE(𝒢, quiet_mosek), Zs)
```



```
plot(support(npmle_fit.prior), probs(npmle_fit.prior), seriestype=:sticks,
     xlab=L"\mu", ylab=L"G(\mu)",label="")
```

```
npmle_postmeans =   [0.135726, 0.250864, 0.435003, 0.68792, 0.815941]
```

```
npmle_postmeans = postmean_targets.(npmle_fit)
```

χ² F-localization intervals

```
chisq_floc =   ChiSquaredFLocalization(0.05)
```

```
chisq_floc = Empirikos.ChiSquaredFLocalization(0.05)
```

```
floc_method_chisq =
EB intervals with F-Localization: ChiSquaredFLocalization{Float64}(0.05)
                  𝒢: DiscretePriorClass | support = 0.0:0.01:5.0
```

```
floc_method_chisq = FLocalizationInterval(flocalization = chisq_floc,
                                       convexclass= 𝒢, solver=quiet_mosek)
```

```
chisq_cis =
  [lower = 0.1317, upper = 0.1398, α = 0.05  (PosteriorMean{PoissonSample{Int64, Float64}}(
```

```
chisq_cis = confint.(floc_method_chisq, postmean_targets, Zs)
```

AMARI intervals

```
discr =   Discretizer([( .. 0], 1, 2, 3, 4, [5 .. )])
```

```
discr = integer_discretizer(0:5)
```

```
amari_chisq =
AMARI with  F-Localization: Empirikos.FittedChiSquaredFLocalization{Any, Any, DataStructur
            𝒢: DiscretePriorClass | support = 0.0:0.01:5.0
```

```
amari_chisq = AMARI(
    flocalization = fit(Empirikos.ChiSquaredFLocalization(0.01), Zs),
    solver=quiet_mosek, convexclass=𝒢, discretizer=discr)
```

```
postmean_ci_amari =
  [lower = 0.1333, upper = 0.1381, α = 0.05  (PosteriorMean{PoissonSample{Int64, Float64}}(
```

```
postmean_ci_amari = confint.(amari_chisq, postmean_targets, Zs)
```

Table of results

	z	N	NPMLE	F_loc_lower	F_loc_upper	Amari_lower	Amari_upper
1	0	103704	0.14	0.13	0.14	0.13	0.14
2	1	14075	0.25	0.23	0.27	0.24	0.26
3	2	1766	0.44	0.36	0.53	0.38	0.49
4	3	255	0.69	0.48	0.94	0.53	0.91
5	4	45	0.82	0.52	1.64	0.58	1.39

```
DataFrame(z=0:4, N=Ns[1:5],
          NPMLE = round.(npmle_postmeans,digits=2),
          F_loc_lower = round.(getproperty.(chisq_cis, :lower), digits=2),
          F_loc_upper = round.(getproperty.(chisq_cis, :upper), digits=2),
          Amari_lower = round.(getproperty.(postmean_ci_amari, :lower), digits=2),
          Amari_upper = round.(getproperty.(postmean_ci_amari, :upper), digits=2)
)
```