AMS-512 Capital Markets and Portfolio Theory

Portfolio Selection (Alternate Risk Measures)

Robert J. Frey, Research Professor
Stony Brook University, Applied Mathematics and Statistics

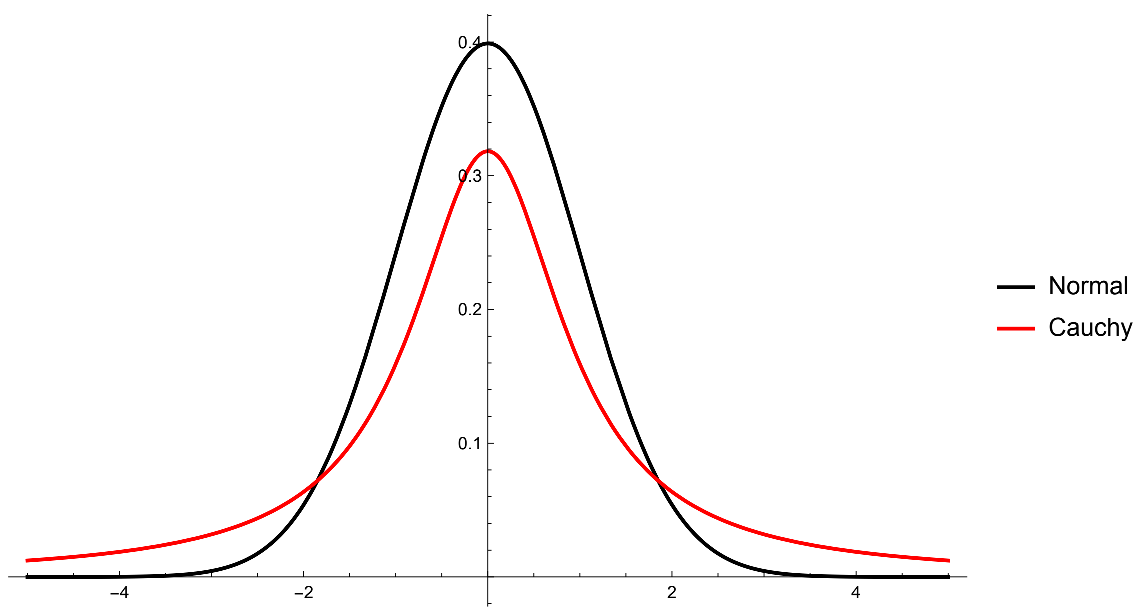
frey@ams.sunysb.edu
http://www.ams.sunysb.edu/~frey

Set Up

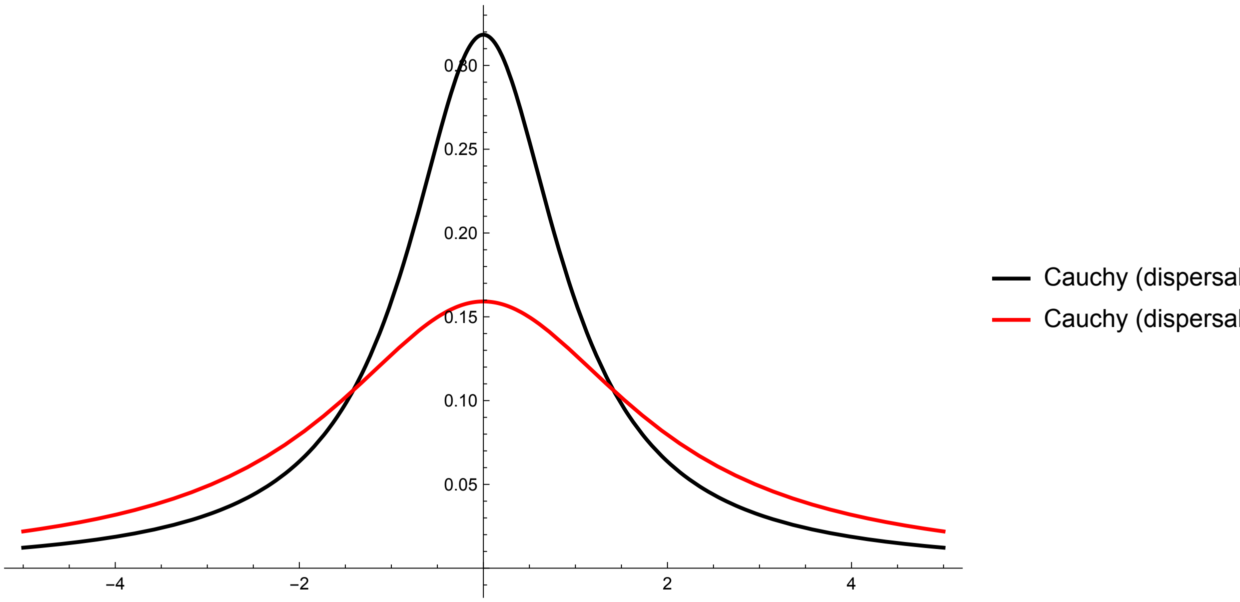
```
Get[FileNameJoin[{NotebookDirectory[], "QuadraticProgramming.m"}]]
```

Normal vs. Cauchy Distributions

```
Plot[Evaluate[{PDF[NormalDistribution[], x], PDF[CauchyDistribution[], x]}],
 {x, -5, 5}, ImageSize → 500, PlotLegends → {"Normal", "Cauchy"},
 PlotStyle → {{Thick, Black}, {Thick, Red}}]
```



```
Plot[Evaluate[{PDF[CauchyDistribution[0, 1], x], PDF[CauchyDistribution[0, 2], x]}],
 {x, -5, 5}, ImageSize → 500,
 PlotLegends → {"Cauchy (dispersal = 1)", "Cauchy (dispersal = 2)"},
 PlotStyle → {{Thick, Black}, {Thick, Red}}]
```




```
PDF[CauchyDistribution[], x]
```

$$\frac{1}{\pi\left(1+x^2\right)}$$

```
PDF[CauchyDistribution[m, d], x]
```

$$\frac{1}{d\,\pi\left(1+\frac{(-m+x)^2}{d^2}\right)}$$

Simulations of Sample σ

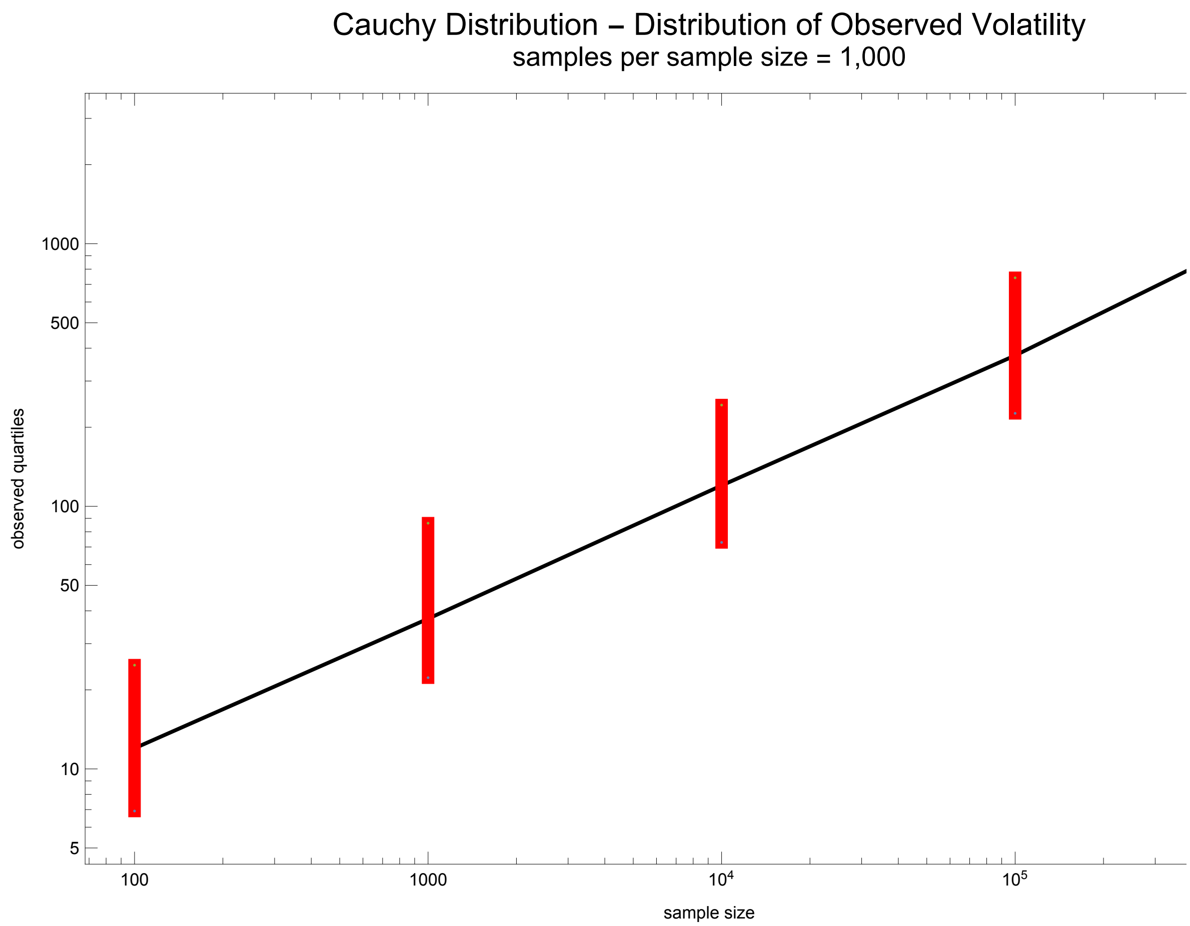
Simulator

```
xSimWithForAt[dist_, xdev_, quan_] := Transpose[
  {{#[[1]], #[[2, 1]]}, {#[[1]], #[[2, 2]]}, {#[[1]], #[[2, 3]]}} & /@
   Table[
    {10^n, {Quantile[#, quan[[1]]], Quantile[#, quan[[2]]], Quantile[#, quan[[3]]]}} &[
     Table[xdev[RandomVariate[dist, 10^n]],
      {1000}
     ]
    ],
    {n, 2, 6}
   ]
  ];
```

Cauchy

```
simCauchySdev =
  xSimWithForAt[CauchyDistribution[], StandardDeviation, {0.25, 0.5, 0.75}];
```

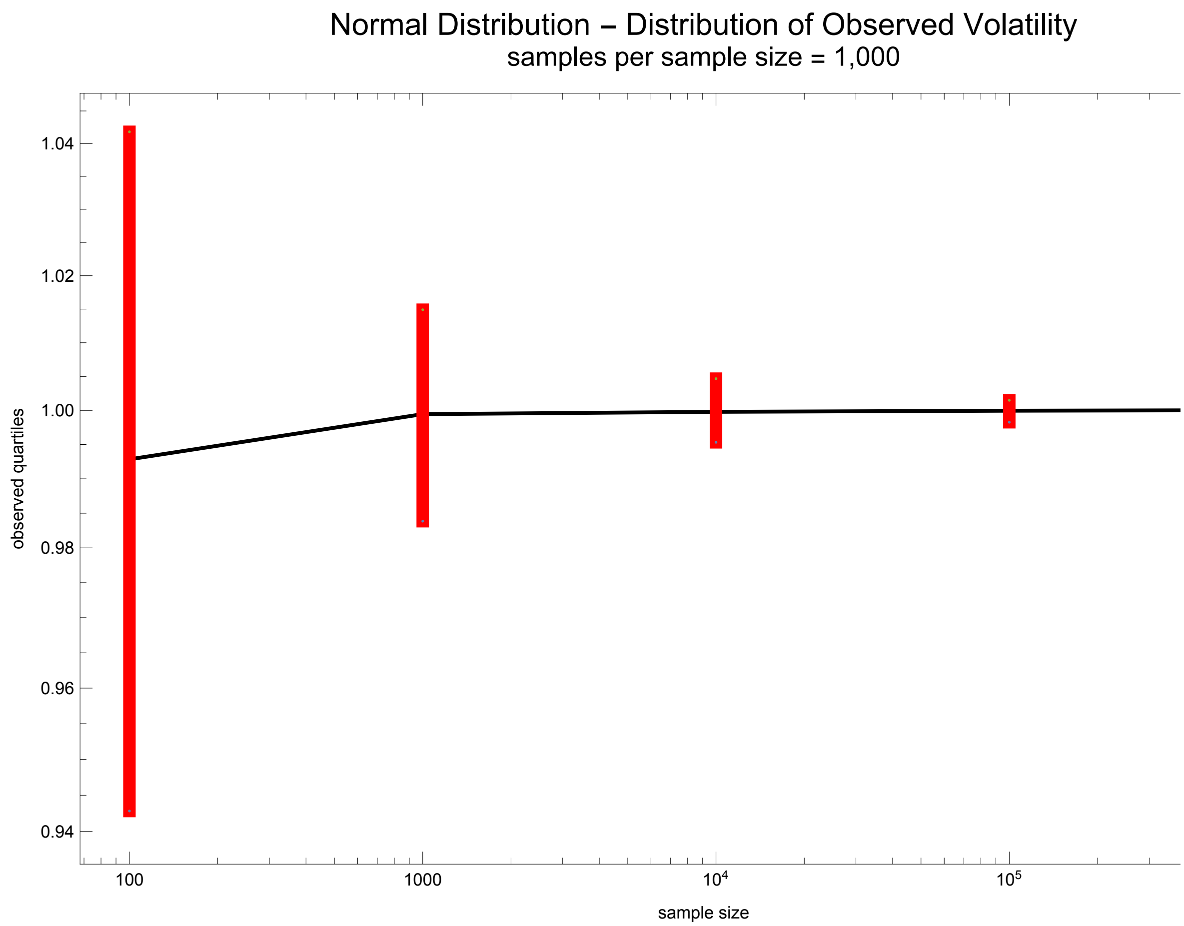
```
gCauchySdev = ListLogLogPlot[simCauchySdev, Filling → {1 → {3}},
  FillingStyle → Directive[Thickness[0.01], Opacity[1, Red]],
  PlotStyle → {{PointSize[0]}, {Black, Thick}, {PointSize[0]}},
  Joined → {False, True, False}, Frame → True,
  FrameLabel → {{"observed quartiles", ""}, {"sample size",
     Column[{Style["Cauchy Distribution - Distribution of Observed Volatility",
        FontSize → 16], Style["samples per sample size = 1,000", FontSize → 14]},
      Center]}}, ImageSize → 750, ImagePadding → {{75, 25}, {50, 75}}]
```



Normal

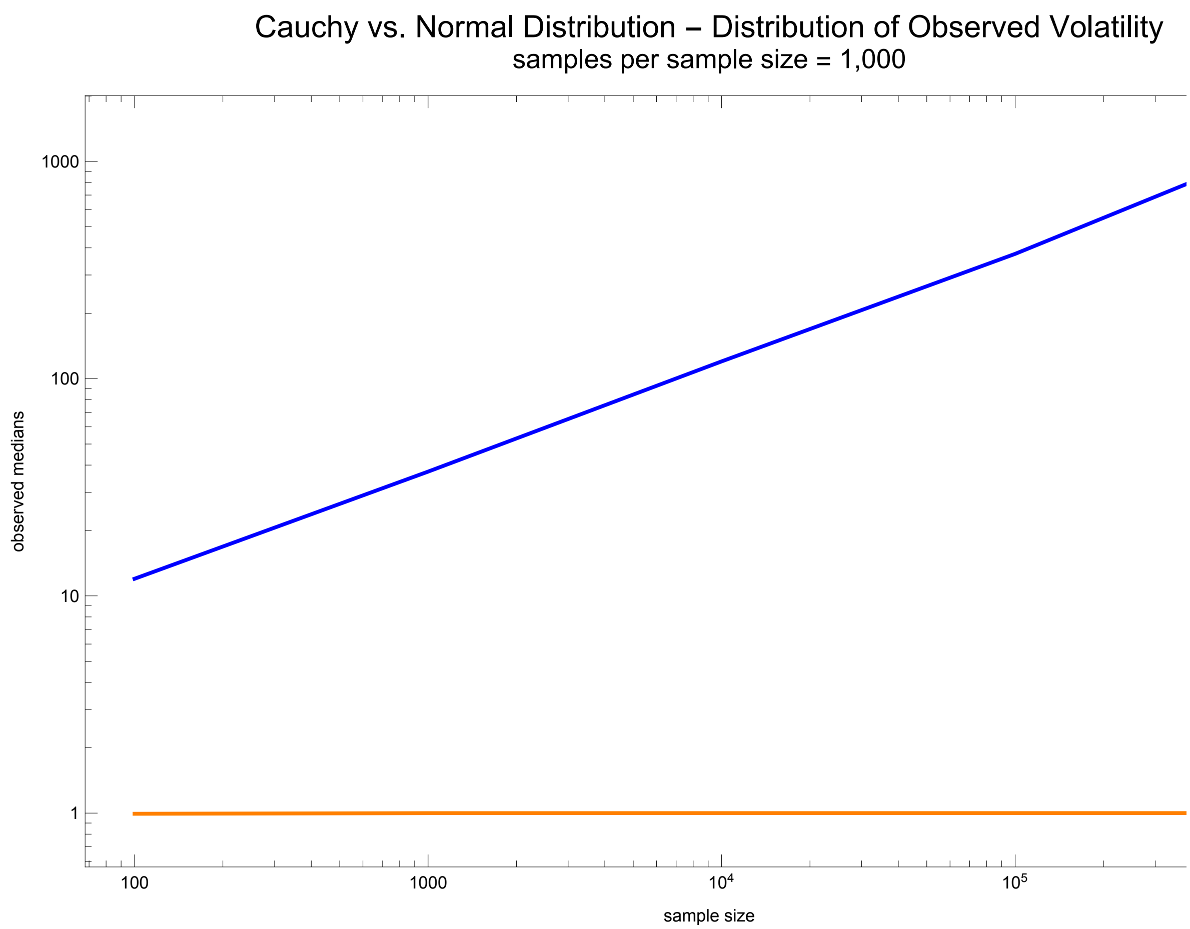
```
simNormalSdev =
  xSimWithForAt[NormalDistribution[], StandardDeviation, {0.25, 0.5, 0.75}];
```

```
gNormalSdev = ListLogLogPlot[simNormalSdev, Filling → {1 → {3}},
  FillingStyle → Directive[Thickness[0.01], Opacity[1, Red]],
  PlotStyle → {{PointSize[0]}, {Black, Thick}, {PointSize[0]}},
  Joined → {False, True, False}, Frame → True,
  FrameLabel → {{"observed quartiles", ""}, {"sample size",
     Column[{Style["Normal Distribution - Distribution of Observed Volatility",
        FontSize → 16], Style["samples per sample size = 1,000", FontSize → 14]},
      Center]}}, ImageSize → 750, ImagePadding → {{75, 25}, {50, 75}}]
```



Comparison

```
gComparisonSdev = ListLogLogPlot[{simCauchySdev⟦2⟧, simNormalSdev⟦2⟧},
  PlotStyle → {{Blue, Thick}, {Orange, Thick}}, Joined → True, Frame → True,
  FrameLabel → {{"observed medians", ""}, {"sample size", Column[{Style[
        "Cauchy vs. Normal Distribution – Distribution of Observed Volatility",
        FontSize → 16], Style["samples per sample size = 1,000",
        FontSize → 14]}, Center]}}, ImageSize → 750,
  ImagePadding → {{75, 25}, {50, 75}}, PlotLegends → {"Cauchy", "Normal"}]
```



The standard deviation is a “known” quantity of distributions in Mathematica. It is defined for the Normal distribution. However, it is indeterminate for the Cauchy distribution; *i.e.*, the integral for $\mathbb{E}\left[(X-\mu_X)^2\right]$ diverges for $X \approx$ Cauchy.

```
StandardDeviation[NormalDistribution[]]
```

1

```
StandardDeviation[CauchyDistribution[]]
```

```
Indeterminate
```

Median Deviation

```
? MedianDeviation
```

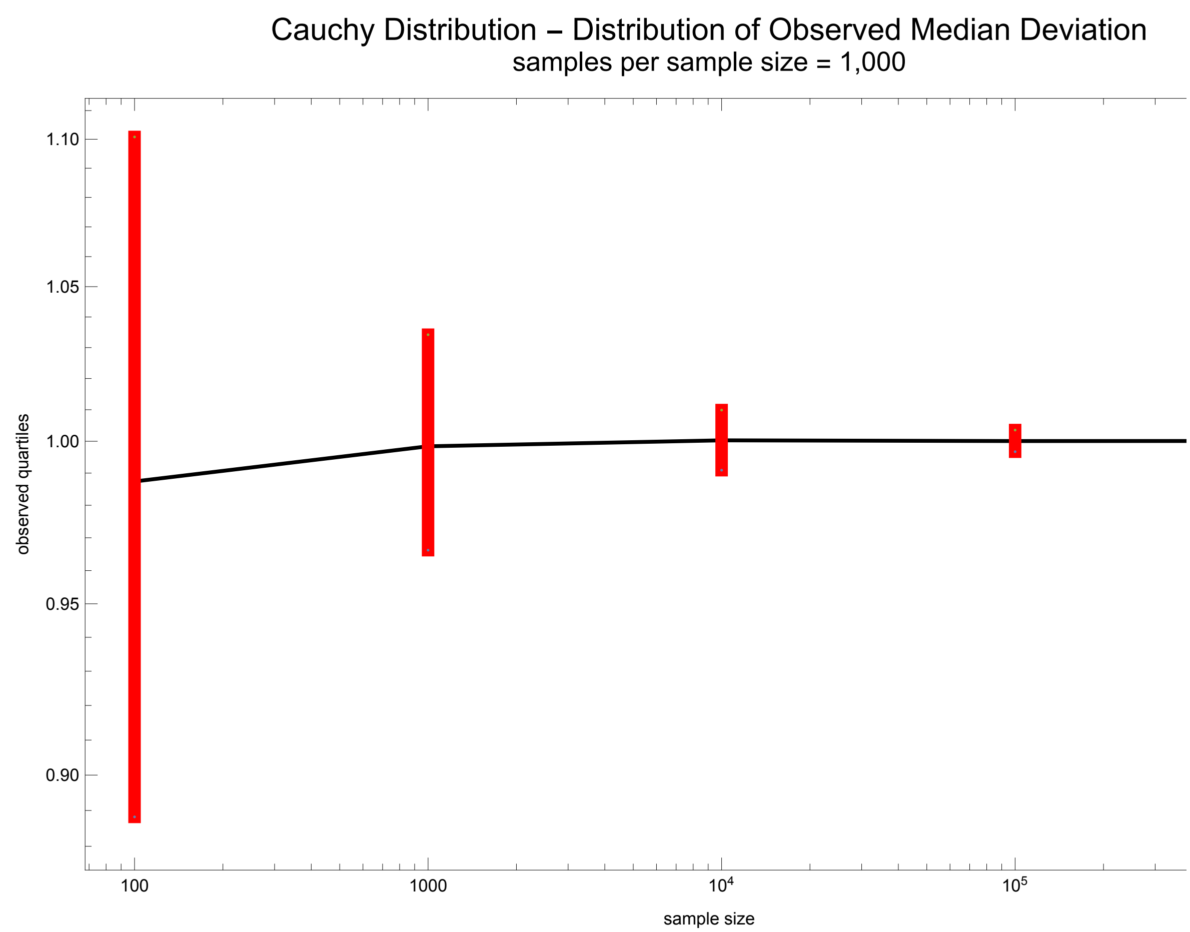
MedianDeviation[*list*] gives the median absolute deviation from the median of the elements in *list*. >>

For the list $\{x_1, x_2, \ldots, x_n\}$, the median deviation is given by the median of the list $\{|x_1 - \tilde{x}|, \ldots, |x_n - \tilde{x}|\}$, where $\tilde{x}$ is the median of the list.

Cauchy

```
simCauchyMdev =
  xSimWithForAt[CauchyDistribution[], MedianDeviation, {0.25, 0.5, 0.75}];
```

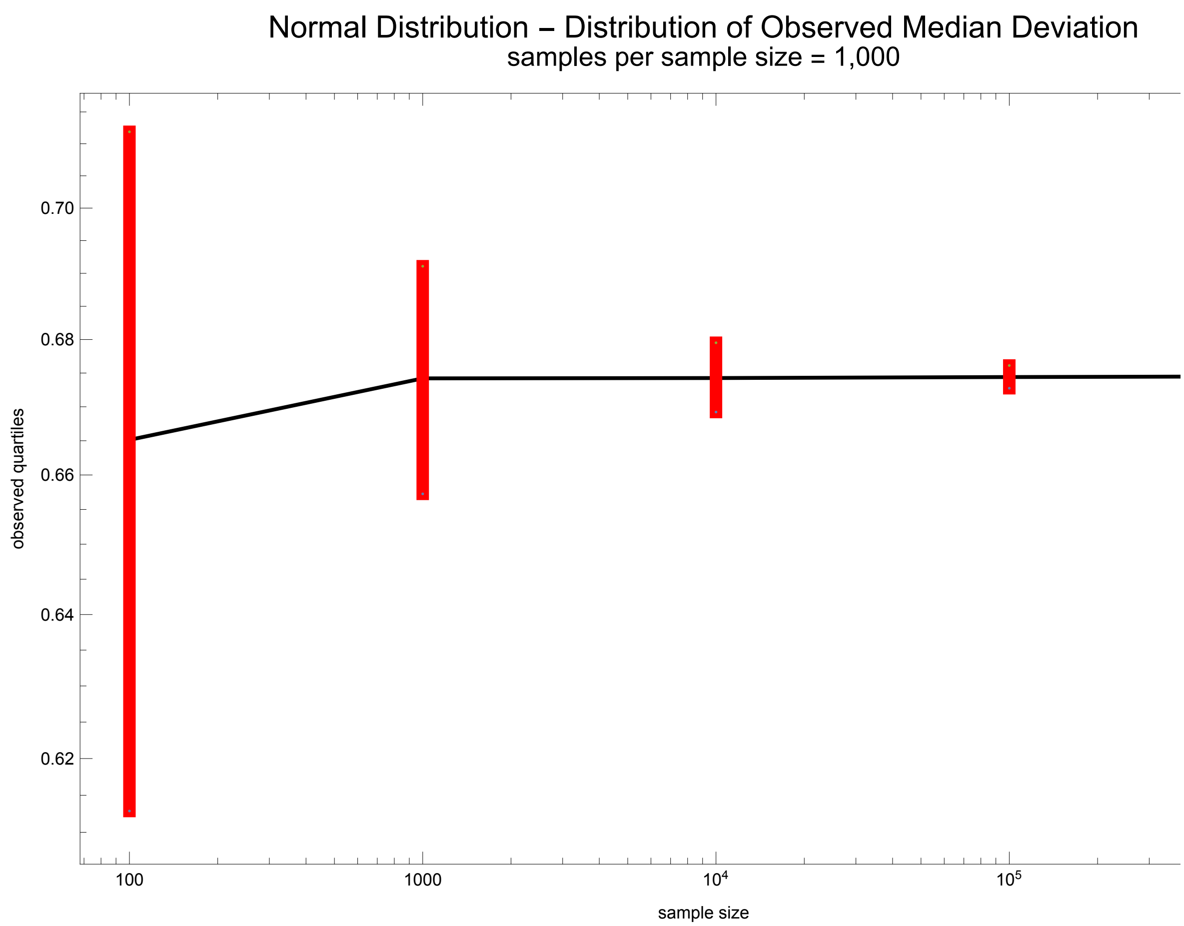
```
gCauchyMdev = ListLogLogPlot[simCauchyMdev, Filling → {1 → {3}},
  FillingStyle → Directive[Thickness[0.01], Opacity[1, Red]],
  PlotStyle → {{PointSize[0]}, {Black, Thick}, {PointSize[0]}},
  Joined → {False, True, False}, Frame → True,
  FrameLabel → {{"observed quartiles", ""}, {"sample size", Column[
      {Style["Cauchy Distribution – Distribution of Observed Median Deviation",
        FontSize → 16], Style["samples per sample size = 1,000", FontSize → 14]},
      Center]}}, ImageSize → 750, ImagePadding → {{75, 25}, {50, 75}}]
```



Normal

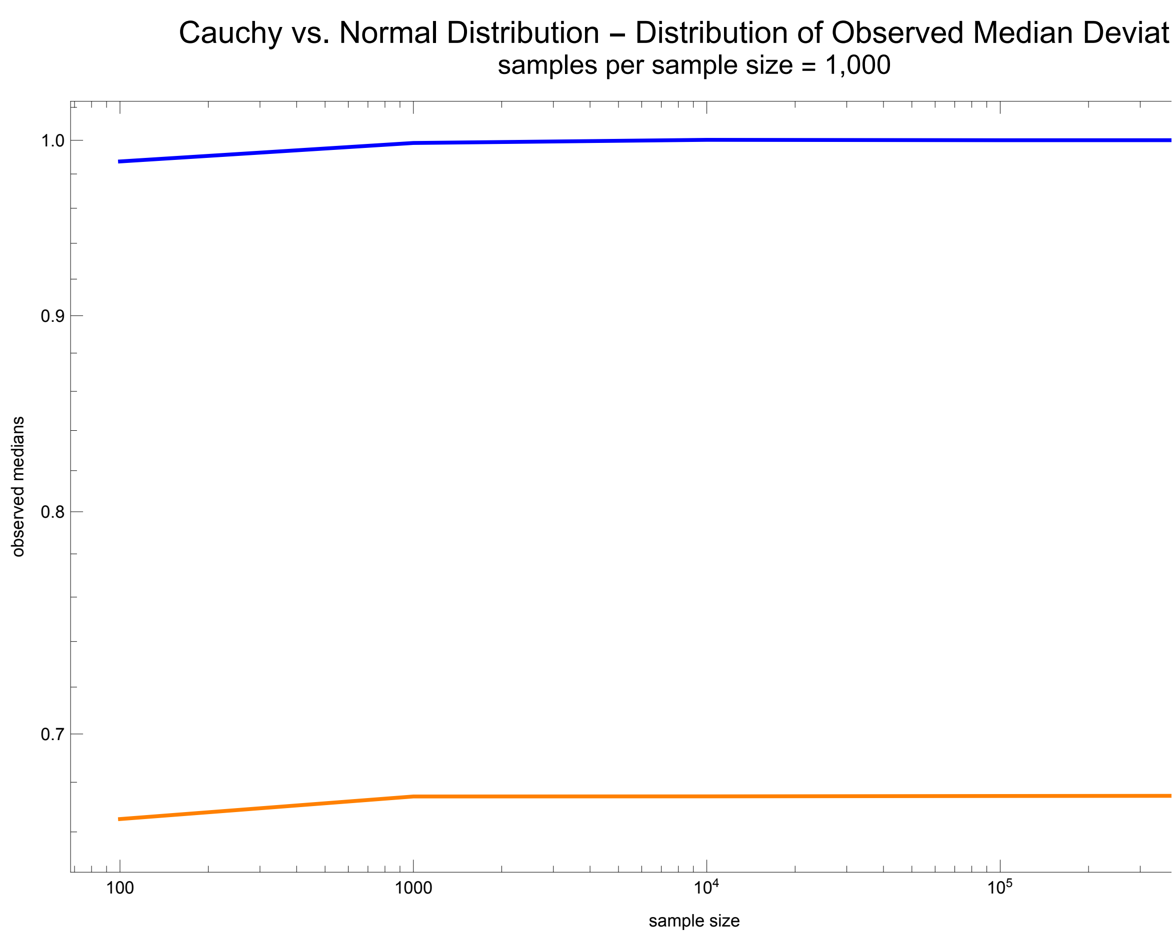
```
simNormalMdev =
  xSimWithForAt[NormalDistribution[], MedianDeviation, {0.25, 0.5, 0.75}];
```

```
gNormalMdev = ListLogLogPlot[simNormalMdev, Filling → {1 → {3}},
  FillingStyle → Directive[Thickness[0.01], Opacity[1, Red]],
  PlotStyle → {{PointSize[0]}, {Black, Thick}, {PointSize[0]}},
  Joined → {False, True, False}, Frame → True,
  FrameLabel → {{"observed quartiles", ""}, {"sample size", Column[
      {Style["Normal Distribution - Distribution of Observed Median Deviation",
        FontSize → 16], Style["samples per sample size = 1,000", FontSize → 14]},
      Center]}}, ImageSize → 750, ImagePadding → {{75, 25}, {50, 75}}]
```



Comparison

```
gComparisonMdev = ListLogLogPlot[{simCauchyMdev⟦2⟧, simNormalMdev⟦2⟧},
  PlotStyle → {{Blue, Thick}, {Orange, Thick}}, Joined → True,
  Frame → True, FrameLabel → {{"observed medians", ""},
    {"sample size", Column[{Style["Cauchy vs. Normal Distribution -
          Distribution of Observed Median Deviation", FontSize → 16],
       Style["samples per sample size = 1,000", FontSize → 14]}, Center]}},
  ImageSize → 750, ImagePadding → {{75, 25}, {50, 75}},
  PlotLegends → {"Cauchy", "Normal"}]
```



The median deviation is not a “known” quantity of distributions in Mathematica, but we can compute it by taking advantage of the fact that both the Normal and Cauchy distributions are unimodal and symmetric by computing the average of the interval between the 25th and 75th percentiles.

```
MedianDeviation[NormalDistribution[]]
```

```
MedianDeviation[NormalDistribution[0, 1]]
```

```
(InverseCDF[NormalDistribution[], 0.75] - InverseCDF[NormalDistribution[], 0.25]) / 2
```

0.67449

```
MedianDeviation[CauchyDistribution[]]
```

MedianDeviation[CauchyDistribution[0, 1]]

```
(InverseCDF[CauchyDistribution[], 0.75] - InverseCDF[CauchyDistribution[], 0.25]) / 2
```

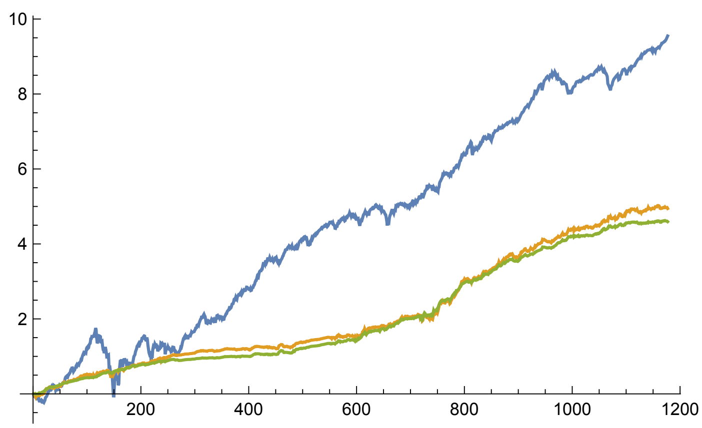
1.

Data

```
mnLogRet = Import[FileNameJoin[{NotebookDirectory[], "mnLogRet.m"}]];
Dimensions[mnLogRet]
```

{1177, 3}

```
ListLinePlot[Accumulate[mnLogRet]ᵀ]
```




```
distClassN = EstimatedDistribution[mnLogRet, MultinormalDistribution[
   {m11, m12, m13}, {{s111, s112, s113}, {s112, s122, s123}, {s113, s123, s133}}]]
```

MultinormalDistribution[{0.00811225, 0.0042015, 0.00390359},
 {{0.00266847, 0.0000853757, 0.0000238696}, {0.0000853757,
   0.000352179, 0.000170079}, {0.0000238696, 0.000170079, 0.00013285}}]

```
ℋN = DistributionFitTest[mnLogRet, distClassN, "TestConclusion"]
```

The null hypothesis that the data is distributed according to the
  MultinormalDistribution[{0.00811225, 0.0042015, 0.00390359},
   {{0.00266847, 0.0000853757, 0.0000238696}, {0.0000853757,
     0.000352179, 0.000170079}, {0.0000238696, 0.000170079, 0.00013285}}]
 is rejected at the 5 percent level based on the Mardia Combined test.

```
distClassT = EstimatedDistribution[mnLogRet, MultivariateTDistribution[
   {m11, m12, m13}, {{s111, s112, s113}, {s112, s122, s123}, {s113, s123, s133}}, d]]
```

```
MultivariateTDistribution[{0.0113767, 0.00309234, 0.00310319},
 {{0.00112147, 0.0000306237, 8.83565 × 10^-6},
  {0.0000306237, 0.000147653, 0.0000707499},
  {8.83565 × 10^-6, 0.0000707499, 0.0000533017}}, 3.00231]
```

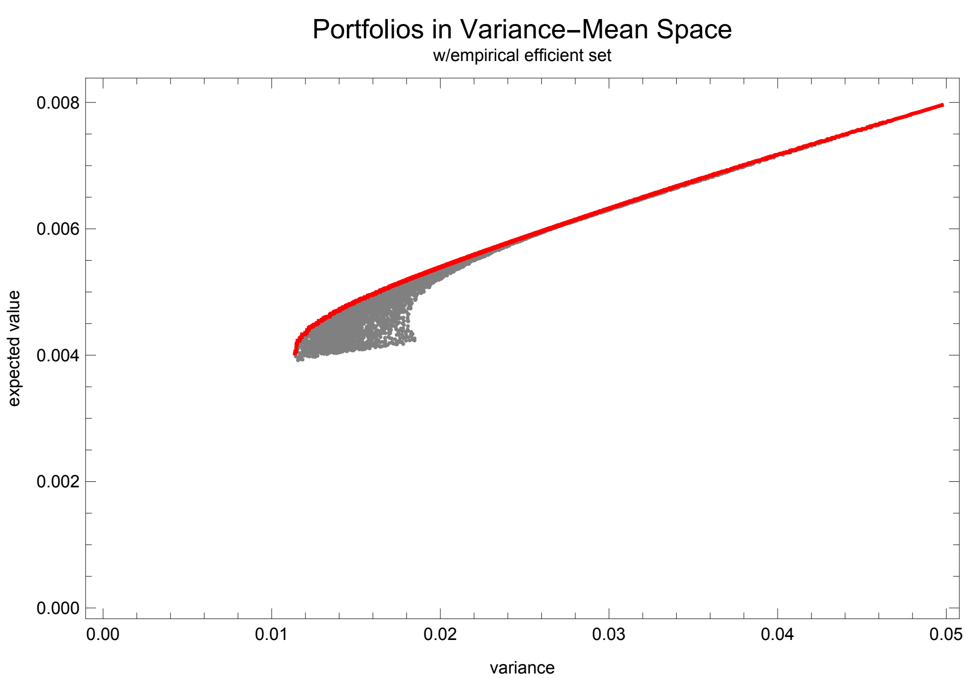
```
ℋT = DistributionFitTest[mnLogRet, distClassT, "TestConclusion"]
```

```
The null hypothesis that the data is distributed according to the
  MultivariateTDistribution[{0.0113767, 0.00309234, 0.00310319},
   {{0.00112147, 0.0000306237, 8.83565 × 10^-6}, {0.0000306237, 0.000147653,
     0.0000707499}, {8.83565 × 10^-6, 0.0000707499, 0.0000533017}}, 3.00231]
 is rejected at the 5 percent level based on the Kolmogorov-Smirnov test.
```

Monte Carlo Simulation

```
sim = Table[
   Through[{StandardDeviation, Mean}[
     mnLogRet.(# / Total[#] &[RandomVariate[UniformDistribution[], 3]])]],
   {10 000}
  ];
```

```
Show[
 ListPlot[
  sim,
  PlotStyle → {Gray, PointSize[Small]},
  PlotRange → All,
  AxesOrigin → {0, 0}
 ],
 ListPlot[
  {First /@#, Rest[FoldList[Max, -∞, Last /@#]]}ᵀ &[Sort[sim]],
  Joined → True,
  PlotStyle → {Thick, Red}
 ],
 Frame → True,
 FrameLabel → {
   "variance",
   "expected value",
   Column[
    {
     Style["Portfolios in Variance-Mean Space", FontSize → 14],
     "w/empirical efficient set"
    },
    Center
   ]},
 ImageSize → 500
]
```



Quadratic Programming

Consider adding a no short constraint to (1).

$$\mathcal{M} = \min_{x} \left\{ \frac{1}{2} \mathbf{x}^T \Sigma \mathbf{x} \;\middle|\; \mu^T \mathbf{x} = r_{\text{targ}} \wedge 1^T \mathbf{x} = 1 \wedge \mathbf{x} \geq \mathbf{0} \right\}$$

We no longer have a direct analytical solution and must resort to numerical techniques. We will use the xQuadraticProgramming[] function.

```
? xQuadraticProgramming
```

xQuadraticProgramming[{lin, quad}, cons, rhs, bound]
 The objective function minimized is "0.5 x.quad.x + lin.x".

```
vnMean = Mean[mnLogRet];
mnCov = Covariance[mnLogRet];
```

```
mnCons = {vnMean, {1, 1, 1}};
xRhs = {{#, 0}, {1, 0}} &;
mnBounds = {{0, ∞}, {0, ∞}, {0, ∞}};
```

```
xDisplayQP["Sample QP", {{"target", "capital"}, {"x1", "x2", "x3"}},
 {{}, mnCov}, mnCons, xRhs[τ], mnBounds]
```

Sample QP

Objective

Lin	{}

Quad

	x1	x2	x3
x1	0.00267074	0.0000854483	0.0000238899
x2	0.0000854483	0.000352478	0.000170224
x3	0.0000238899	0.000170224	0.000132963

Constraints

	x1	x2	x3		RHS
target	0.00811225	0.0042015	0.00390359	==	τ
capital	1	1	1	==	1

Bounds

	LO	UP
x1	0	∞
x2	0	∞
x3	0	∞

```
xQuadraticProgramming[{{}, mnCov}, {{1, 1, 1}}, {{1, 0}}, mnBounds]
```

```
{0.0000644374, {0.0423887, 0.00269248, 0.954919}}
```

```
nMinVarRet = vnMean.Last[%]
```

```
0.00408279
```

```
xQuadraticProgramming[{-vnMean, {}}, {{1, 1, 1}}, {{1, 0}}, mnBounds]
```

```
{-0.00811225, {1., 0., 0.}}
```

```
nMaxRet = Max[vnMean]
```

0.00811225

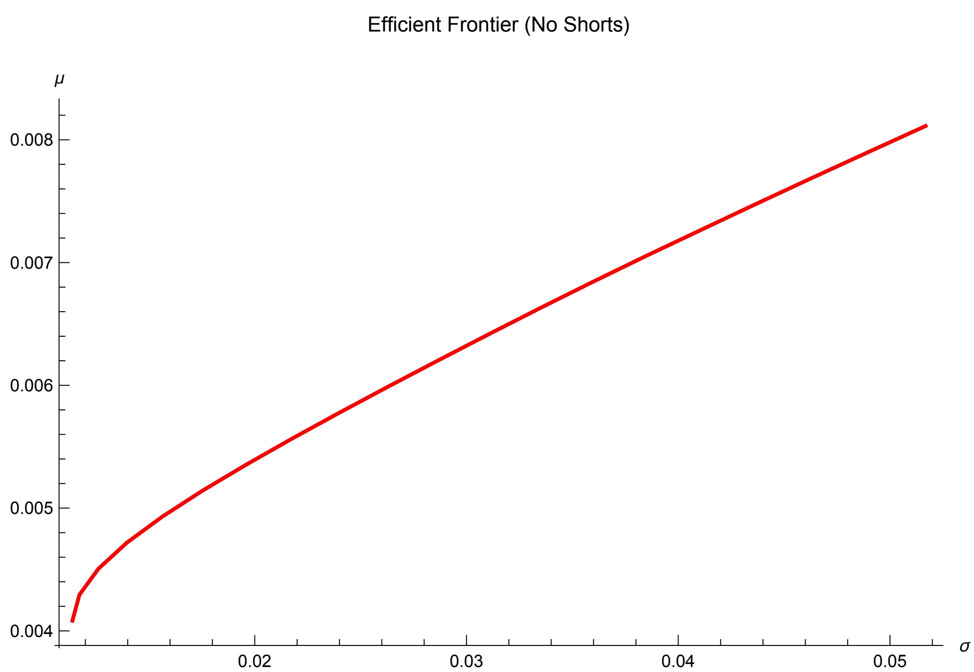
```
mnNoShortEffPort = Table[
   Last[xQuadraticProgramming[{{}, mnCov}, mnCons, xRhs[r], mnBounds]],
   {r, nMinVarRet, nMaxRet, (nMaxRet - nMinVarRet)/19}
  ];
TableForm[mnNoShortEffPort, TableHeadings → Automatic]
```

	1	2	3
1	0.0416427	0.0132311	0.945126
2	0.0918928	0.0152158	0.892891
3	0.141829	0.0216355	0.836536
4	0.191546	0.0311449	0.777309
5	0.240872	0.0461894	0.712939
6	0.289603	0.0696404	0.640757
7	0.335978	0.126359	0.537663
8	0.38405	0.159122	0.456829
9	0.430036	0.221344	0.34862
10	0.477769	0.258882	0.263348
11	0.525758	0.292811	0.18143
12	0.572379	0.346065	0.0815558
13	0.622806	0.345559	0.0316353
14	0.675463	0.313531	0.0110057
15	0.729162	0.266791	0.00404655
16	0.784099	0.202561	0.0133393
17	0.837714	0.157009	0.00527625
18	0.891612	0.107472	0.000916887
19	0.945827	0.0534414	0.000732034
20	1.	0.	0.

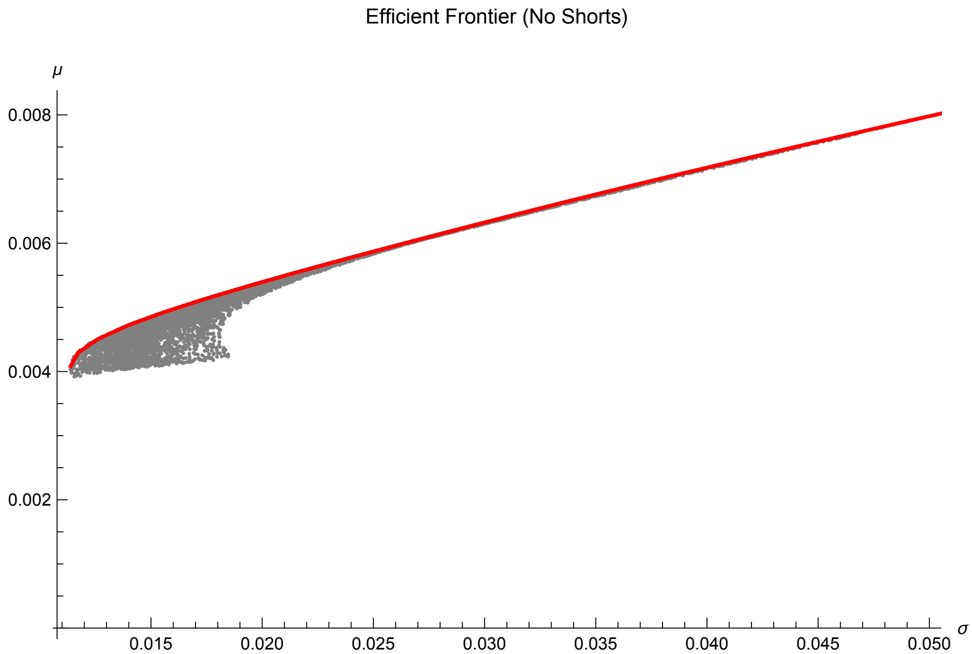
```
mnNoShortEffFront = {√#.mnCov.# , vnMean.#} & /@ mnNoShortEffPort;
TableForm[mnNoShortEffFront, TableHeadings → Automatic]
```

	1	2
1	0.0113884	0.00408279
2	0.0117228	0.00429487
3	0.012621	0.00450695
4	0.0139658	0.00471902
5	0.0156399	0.0049311
6	0.0175442	0.00514318
7	0.0196101	0.00535525
8	0.0217753	0.00556733
9	0.0240215	0.00577941
10	0.0263218	0.00599148
11	0.0286671	0.00620356
12	0.0310446	0.00641564
13	0.0334606	0.00662771
14	0.0359293	0.00683979
15	0.0384554	0.00705187
16	0.0410494	0.00726394
17	0.0436566	0.00747602
18	0.0462999	0.0076881
19	0.0489786	0.00790017
20	0.0516792	0.00811225

```
ListPlot[mnNoShortEffFront, Joined → True,
 PlotStyle → {Red, Thick}, PlotLabel → "Efficient Frontier (No Shorts)\n",
 AxesLabel → {"σ", "μ"}, PlotRange → All, ImageSize → 500]
```



```
Show[ListPlot[sim, PlotStyle → {Gray, PointSize[Small]},
  PlotLabel → "Efficient Frontier (No Shorts)\n",
  AxesLabel → {"σ", "μ"}, PlotRange → All, ImageSize → 500],
 ListPlot[mnNoShortEffFront, Joined → True, PlotStyle → {Red, Thick}]
]
```



VaR and CVaR

```
xMoments[ret_, port_] :=
 Through[{Mean, StandardDeviation, Skewness, Kurtosis}[ret.port]]

xVaRandCVaR[ret_, port_, dist_, conf_] := Block[
  {cvar, d, h, m, prtn, s, var, μ, σ},
  prtn = ret.port;
  μ = Mean[prtn];
  σ = StandardDeviation[prtn];
  d = EstimatedDistribution[ret.port, dist];
  h = DistributionFitTest[prtn, d, "ShortTestConclusion"];
  var = InverseCDF[d, 1 - conf];
  cvar = NIntegrate[r PDF[d, r], {r, -∞, var}] / (1 - conf);
  {σ, μ, var, cvar, d, h}
 ]
```

```
mnMoments = xMoments[mnLogRet, #] & /@ mnNoShortEffPort;
TableForm[mnMoments, TableHeadings → {Automatic, {"μ", "σ", "sk", "ku"}}]
```

	μ	σ	sk	ku
1	0.00408279	0.0113884	0.737788	12.0638
2	0.00429487	0.0117228	0.647093	10.2832
3	0.00450695	0.012621	0.474686	8.61619
4	0.00471902	0.0139658	0.277583	8.01494
5	0.0049311	0.0156399	0.0991228	8.2659
6	0.00514318	0.0175442	-0.0460917	8.86651
7	0.00535525	0.0196101	-0.162442	9.4489
8	0.00556733	0.0217753	-0.250746	10.0177
9	0.00577941	0.0240215	-0.323045	10.4318
10	0.00599148	0.0263218	-0.37707	10.8026
11	0.00620356	0.0286671	-0.41929	11.0982
12	0.00641564	0.0310446	-0.455828	11.3034
13	0.00662771	0.0334606	-0.478249	11.5225
14	0.00683979	0.0359293	-0.491592	11.7182
15	0.00705187	0.0384554	-0.499667	11.8741
16	0.00726394	0.0410494	-0.50324	11.9958
17	0.00747602	0.0436566	-0.507682	12.0729
18	0.0076881	0.0462999	-0.51048	12.1267
19	0.00790017	0.0489786	-0.511962	12.162
20	0.00811225	0.0516792	-0.512957	12.1821

```
mnRiskN =
  xVaRandCVaR[mnLogRet, #, NormalDistribution[m, s], 0.995] & /@ mnNoShortEffPort;
TableForm[mnRiskN, TableHeadings →
  {Automatic, {"σ", "μ", "VaR", "CVaR", "dist", "test"}}]
```

	σ	μ	VaR	CVaR	dist
1	0.0113884	0.00408279	-0.0252393	-0.0288379	NormalDistribution[0.004
2	0.0117228	0.00429487	-0.0258882	-0.0295924	NormalDistribution[0.004
3	0.012621	0.00450695	-0.0279887	-0.0319768	NormalDistribution[0.004
4	0.0139658	0.00471902	-0.0312392	-0.0356522	NormalDistribution[0.004
5	0.0156399	0.0049311	-0.0353375	-0.0402794	NormalDistribution[0.004
6	0.0175442	0.00514318	-0.0400284	-0.0455721	NormalDistribution[0.005
7	0.0196101	0.00535525	-0.0451356	-0.0513322	NormalDistribution[0.005
8	0.0217753	0.00556733	-0.0504984	-0.0573791	NormalDistribution[0.005
9	0.0240215	0.00577941	-0.0560695	-0.06366	NormalDistribution[0.005
10	0.0263218	0.00599148	-0.0617801	-0.0700974	NormalDistribution[0.005
11	0.0286671	0.00620356	-0.0676065	-0.0766649	NormalDistribution[0.006
12	0.0310446	0.00641564	-0.0735161	-0.0833257	NormalDistribution[0.006
13	0.0334606	0.00662771	-0.0795244	-0.0900974	NormalDistribution[0.006
14	0.0359293	0.00683979	-0.0856686	-0.0970217	NormalDistribution[0.006
15	0.0384554	0.00705187	-0.0919605	-0.104112	NormalDistribution[0.007
16	0.0410494	0.00726394	-0.0984273	-0.111398	NormalDistribution[0.007
17	0.0436566	0.00747602	-0.104928	-0.118723	NormalDistribution[0.007
18	0.0462999	0.0076881	-0.111522	-0.126152	NormalDistribution[0.007
19	0.0489786	0.00790017	-0.118207	-0.133683	NormalDistribution[0.007
20	0.0516792	0.00811225	-0.124948	-0.141278	NormalDistribution[0.008

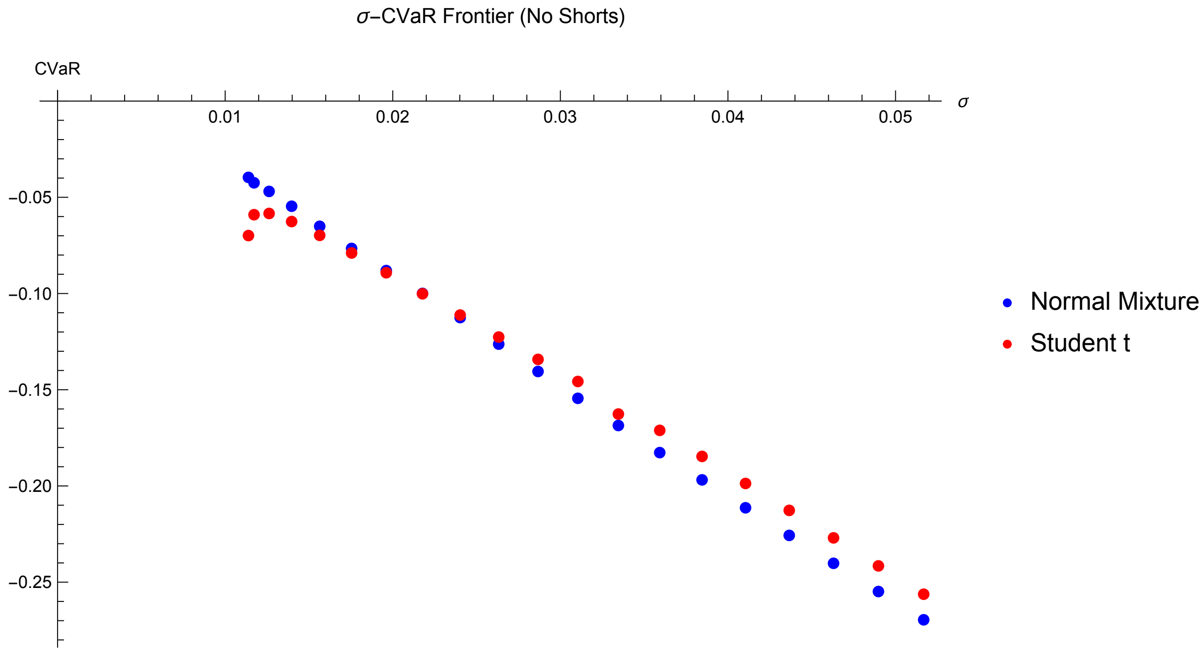
```
mnRiskT = xVaRandCVaR[mnLogRet, #, StudentTDistribution[m, s, d], 0.995] & /@
   mnNoShortEffPort;
TableForm[mnRiskT, TableHeadings →
  {Automatic, {"σ", "μ", "VaR", "CVaR", "dist", "test"}}]
```

	σ	μ	VaR	CVaR	dist
1	0.0113884	0.00408279	-0.0413278	-0.069892	StudentTDistribution[0.0
2	0.0117228	0.00429487	-0.0381214	-0.059048	StudentTDistribution[0.0
3	0.012621	0.00450695	-0.0392615	-0.0584054	StudentTDistribution[0.0
4	0.0139658	0.00471902	-0.0427707	-0.0626113	StudentTDistribution[0.0
5	0.0156399	0.0049311	-0.047811	-0.0697835	StudentTDistribution[0.0
6	0.0175442	0.00514318	-0.0539013	-0.0789487	StudentTDistribution[0.0
7	0.0196101	0.00535525	-0.0606389	-0.08921	StudentTDistribution[0.0
8	0.0217753	0.00556733	-0.0677434	-0.100145	StudentTDistribution[0.0
9	0.0240215	0.00577941	-0.0750299	-0.111193	StudentTDistribution[0.0
10	0.0263218	0.00599148	-0.0825147	-0.122622	StudentTDistribution[0.0
11	0.0286671	0.00620356	-0.0901296	-0.134229	StudentTDistribution[0.0
12	0.0310446	0.00641564	-0.0977717	-0.14571	StudentTDistribution[0.0
13	0.0334606	0.00662771	-0.107453	-0.162636	StudentTDistribution[0.0
14	0.0359293	0.00683979	-0.114016	-0.171112	StudentTDistribution[0.0
15	0.0384554	0.00705187	-0.122557	-0.184648	StudentTDistribution[0.0
16	0.0410494	0.00726394	-0.13139	-0.198704	StudentTDistribution[0.0
17	0.0436566	0.00747602	-0.140248	-0.212692	StudentTDistribution[0.0
18	0.0462999	0.0076881	-0.149281	-0.226974	StudentTDistribution[0.0
19	0.0489786	0.00790017	-0.158483	-0.24154	StudentTDistribution[0.0
20	0.0516792	0.00811225	-0.167789	-0.256256	StudentTDistribution[0.0

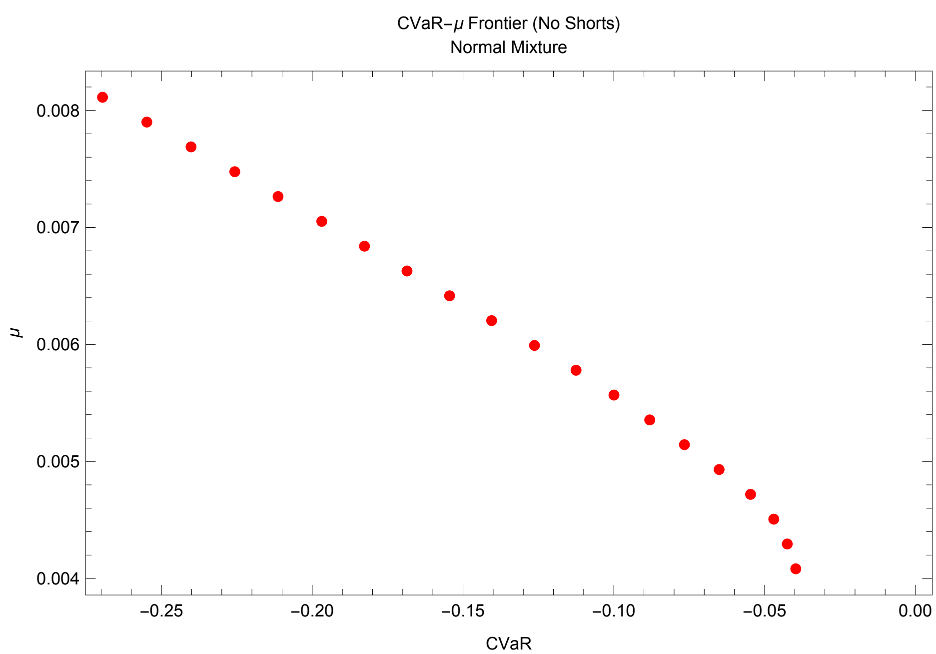
```
mnRiskNM = Block[
   {m1, m2, s1, s2, w1, w2},
   xVaRandCVaR[mnLogRet, #, MixtureDistribution[{w1, w2}, {NormalDistribution[m1,
         s1], NormalDistribution[m2, s2]}], 0.995] & /@ mnNoShortEffPort
  ];
TableForm[mnRiskNM,
 TableHeadings → {Automatic, {"σ", "μ", "VaR", "CVaR", "dist", "test"}}]
```

	σ	μ	VaR	CVaR	dist
1	0.0113884	0.00408279	-0.0333016	-0.039648	MixtureDistribution[{0.6
2	0.0117228	0.00429487	-0.0352072	-0.0424708	MixtureDistribution[{0.7
3	0.012621	0.00450695	-0.0385096	-0.0469448	MixtureDistribution[{0.7
4	0.0139658	0.00471902	-0.0444632	-0.0546437	MixtureDistribution[{0.8
5	0.0156399	0.0049311	-0.0524605	-0.0650662	MixtureDistribution[{0.8
6	0.0175442	0.00514318	-0.0615882	-0.0765747	MixtureDistribution[{0.8
7	0.0196101	0.00535525	-0.0710741	-0.0880895	MixtureDistribution[{0.8
8	0.0217753	0.00556733	-0.0809568	-0.0999444	MixtureDistribution[{0.8
9	0.0240215	0.00577941	-0.0912349	-0.112518	MixtureDistribution[{0.8
10	0.0263218	0.00599148	-0.102287	-0.126302	MixtureDistribution[{0.8
11	0.0286671	0.00620356	-0.113692	-0.14051	MixtureDistribution[{0.8
12	0.0310446	0.00641564	-0.125004	-0.154461	MixtureDistribution[{0.9
13	0.0334606	0.00662771	-0.136544	-0.168579	MixtureDistribution[{0.9
14	0.0359293	0.00683979	-0.148132	-0.182678	MixtureDistribution[{0.9
15	0.0384554	0.00705187	-0.159775	-0.196843	MixtureDistribution[{0.9
16	0.0410494	0.00726394	-0.171634	-0.21134	MixtureDistribution[{0.9
17	0.0436566	0.00747602	-0.183384	-0.22569	MixtureDistribution[{0.9
18	0.0462999	0.0076881	-0.195235	-0.240198	MixtureDistribution[{0.9
19	0.0489786	0.00790017	-0.207188	-0.254855	MixtureDistribution[{0.9
20	0.0516792	0.00811225	-0.219169	-0.269552	MixtureDistribution[{0.9

```
ListPlot[{mnRiskNM⟦All, {1, 4}⟧, mnRiskT⟦All, {1, 4}⟧},
 PlotStyle → {{Blue, Thick}, {Red, Thick}},
 PlotLabel → "σ-CVaR Frontier (No Shorts)\n",
 AxesLabel → {"σ", "CVaR"}, PlotRange → All, AxesOrigin → {0, 0},
 PlotLegends → {"Normal Mixture", "Student t"}, ImageSize → 500]
```




```
ListPlot[mnRiskNM⟦All, {4, 2}⟧, PlotStyle → {Red, Thick},
 FrameLabel → {{"μ", ""}, {"CVaR", "CVaR-μ Frontier (No Shorts)\nNormal Mixture"}},
 Frame → True, PlotRange → All, AxesOrigin → {0, 0}, ImageSize → 500]
```



```
LinearModelFit[mnRiskNM[[All, {1, 4}]], x, x]
```

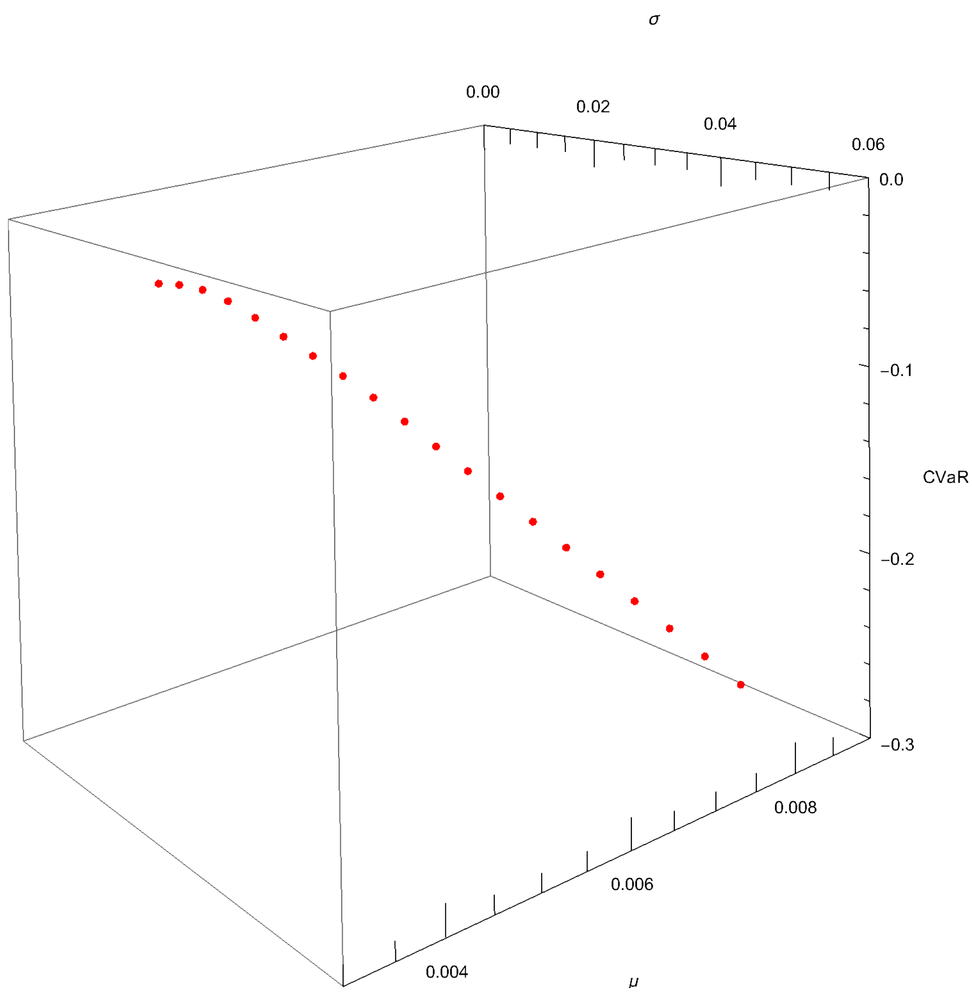
FittedModel[0.0245521 – 5.72982 x]

```
LinearModelFit[mnRiskT[[All, {1, 4}]], x, x]
```

FittedModel[0.00407881 – 4.94559 x]

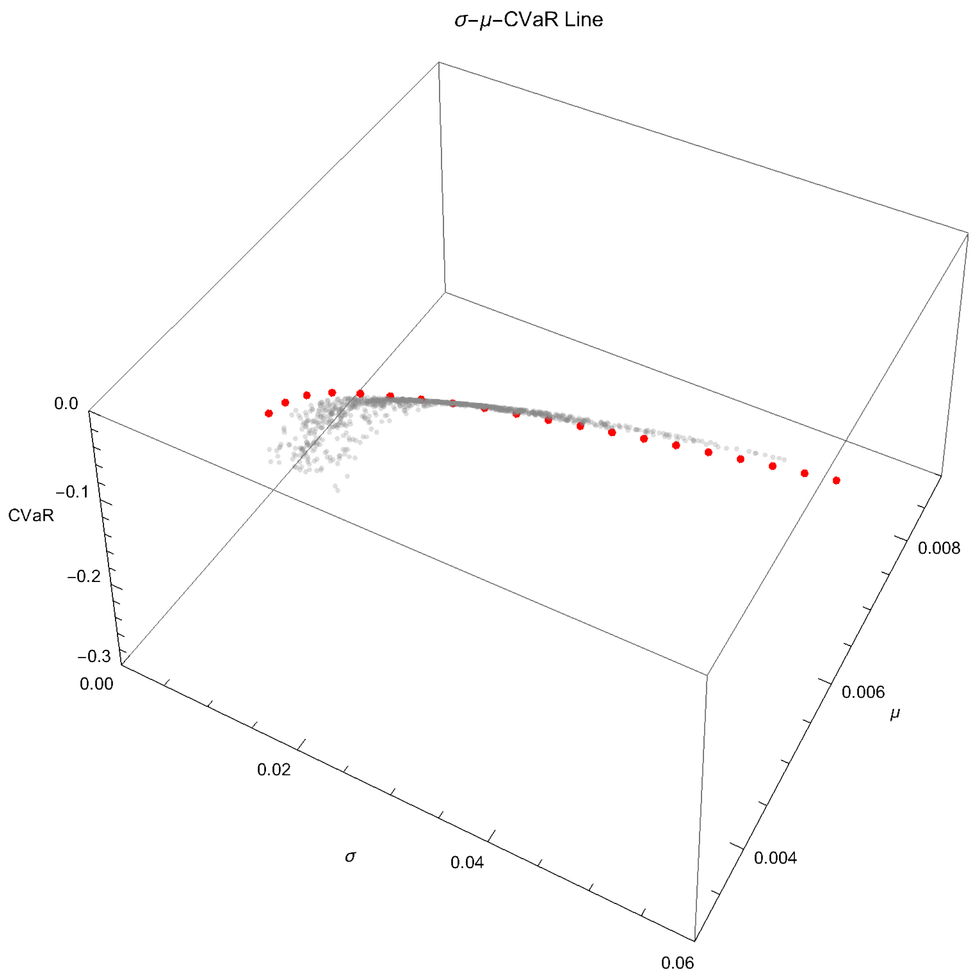
```
ListPointPlot3D[mnRiskNM[[All, {1, 2, 4}]],
 PlotLabel → "σ-μ-CVaR Line\n", AxesLabel → {"σ", "μ", "CVaR"},
 ImageSize → 500, PlotStyle → {Red, PointSize[Medium]},
 AspectRatio → 1, PlotRange → {{0.0, 0.06}, {0.003, 0.009}, {0, -0.3}}]
```

σ–μ–CVaR Line




```
sim3D = Table[
   xVaRandCVaR[mnLogRet, #/Total[#] &[RandomVariate[UniformDistribution[], 3]],
    StudentTDistribution[m, s, d], 0.995],
   {1000}
  ];
```

```
Show[ListPointPlot3D[mnRiskNM[[All, {1, 2, 4}]],
  PlotLabel → "σ-μ-CVaR Line\n", AxesLabel → {"σ", "μ", "CVaR"},
  PlotStyle → {Red, PointSize[Medium]}, ImageSize → 500,
  AspectRatio → 1, PlotRange → {{0.0, 0.06}, {0.003, 0.009}, {0, -0.3}}],
 ListPointPlot3D[sim3D[[All, {1, 2, 4}]], PlotStyle → {Opacity[0.25, Gray]}]]
```



```
Show[ListPointPlot3D[mnRiskT⟦All, {1, 2, 4}⟧,
  PlotLabel → "σ-μ-CVaR Line\n", AxesLabel → {"σ", "μ", "CVaR"},
  PlotStyle → {Red, PointSize[Medium]}, ImageSize → 500,
  AspectRatio → 1, PlotRange → {{0.0, 0.06}, {0.003, 0.009}, {0, -0.3}}],
 ListPointPlot3D[sim3D⟦All, {1, 2, 4}⟧, PlotStyle → {Opacity[0.25, Gray]}]]
```