NES Family

for Mathematica

Jan Koutnik, IDSIA, E–mail: hkou.idsia@ch

Pseudocodes are taken from the NES paper :

- Natural Evolution Strategies.Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun and Jürgen Schmidhuber.Under review in *Journal of Machine Learning Research*.

Note : according to Yi Sun, A is a log covariance matrix in xNES implementation.

---

sNES (Separable NES)

Algorithm 7: Separable NES (SNES)

input: f, $\boldsymbol{\mu}_{init}$, $\boldsymbol{\sigma}_{init}$
repeat
  for $k = 1 \ldots \lambda$ **do**
    draw sample $\mathbf{s}_k \sim \mathcal{N}(0, \mathbb{I})$
    $\mathbf{z}_k \leftarrow \boldsymbol{\mu} + \boldsymbol{\sigma}\mathbf{s}_k$
    evaluate the fitness $f(\mathbf{z}_k)$
  end
  sort $\{(\mathbf{s}_k, \mathbf{z}_k)\}$ with respect to $f(\mathbf{z}_k)$ and compute utilities u_k
  compute gradients $\nabla_{\boldsymbol{\mu}} J \leftarrow \sum_{k=1}^{\lambda} u_k \cdot \mathbf{s}_k$
    $\nabla_{\boldsymbol{\sigma}} J \leftarrow \sum_{k=1}^{\lambda} u_k \cdot (\mathbf{s}_k^2 - 1)$
  update parameters $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \eta_{\boldsymbol{\mu}} \cdot \boldsymbol{\sigma} \cdot \nabla_{\boldsymbol{\mu}} J$
    $\boldsymbol{\sigma} \leftarrow \boldsymbol{\sigma} \cdot \exp(\eta_{\boldsymbol{\sigma}}/2 \cdot \nabla_{\boldsymbol{\sigma}} J)$
until *stopping criterion is met*;

```
sNESstep[f_, dim_, μσ_, λ_, ηδ_, ησ_] := Module[{μ, σ, z, s, u, gδ, gσ},
  {μ, σ} = μσ;
  s = RandomReal[NormalDistribution[0, 1], {λ, dim}];
  s = s[[Ordering[f /@ ((μ + σ sᵀ)ᵀ)]]];
  u = utilityFunction[λ];
  {μ + ηδ σ (u.s), σ Exp[ησ / 2 (u.(s^2 - 1))]}
 ]
```

```
sNES[f_, dim_, μσ_, λ_, ηδ_, ησ_, nIter_] := Nest[sNESstep[f, dim, #, λ, ηδ, ησ] &, μσ, nIter]
```

```
populationSize[d_] := 4 + Floor[3 Log[d]]
```

```
utilityFunction[n_] :=
 utilityFunction[n] = Reverse[N[Max[0, #] & /@ (Log[n / 2 - 1] - Log[Range[n]]) /
      Total[Max[0, #] & /@ (Log[n / 2 - 1] - Log[Range[n]])] - 1 / n]]
```

```
learningRateSNES[d_] := N[(3 + Log[d]) / (5 Sqrt[d])]
```

xNES (Exponential NES)

Algorithm 4: Exponential Natural Evolution Strategies (xNES), for multinormal distributions

input: f, $\boldsymbol{\mu}_{init}$, $\boldsymbol{\Sigma}_{init} = \mathbf{A}^\top\mathbf{A}$

initialize $\quad \sigma \leftarrow \sqrt[d]{|\det(\mathbf{A})|}$
$\quad\quad\quad\quad\;\; \mathbf{B} \leftarrow \mathbf{A}/\sigma$

repeat

  for $k = 1 \ldots \lambda$ **do**

    draw sample $\mathbf{s}_k \sim \mathcal{N}(0, \mathbb{I})$

    $\mathbf{z}_k \leftarrow \boldsymbol{\mu} + \sigma\mathbf{B}^\top\mathbf{s}_k$

    evaluate the fitness $f(\mathbf{z}_k)$

  end

  sort $\{(\mathbf{s}_k, \mathbf{z}_k)\}$ with respect to $f(\mathbf{z}_k)$ and compute utilities u_k

  compute gradients $\quad \nabla_\delta J \leftarrow \sum_{k=1}^{\lambda} u_k \cdot \mathbf{s}_k \quad\quad \nabla_\mathbf{M} J \leftarrow \sum_{k=1}^{\lambda} u_k \cdot (\mathbf{s}_k\mathbf{s}_k^\top - \mathbb{I})$
  $\quad\quad\quad\quad\quad\quad\quad\quad\quad \nabla_\sigma J \leftarrow \mathrm{tr}(\nabla_\mathbf{M} J)/d \quad\quad \nabla_\mathbf{B} J \leftarrow \nabla_\mathbf{M} J - \nabla_\sigma J \cdot \mathbb{I}$

  update parameters $\quad \boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \eta_\delta \cdot \sigma\mathbf{B} \cdot \nabla_\delta J$
  $\quad\quad\quad\quad\quad\quad\quad\quad\quad \sigma \leftarrow \sigma \cdot \exp(\eta_\sigma/2 \cdot \nabla_\sigma J)$
  $\quad\quad\quad\quad\quad\quad\quad\quad\quad \mathbf{B} \leftarrow \mathbf{B} \cdot \exp(\eta_\mathbf{B}/2 \cdot \nabla_\mathbf{B} J)$

until *stopping criterion is met*

```
eye[n_] := eye[n] = IdentityMatrix[n]

xNESstep[f_, dim_, μA_, λ_, ημ_, ηA_] := Module[{μ, σ, z, x, u, gμ, gσ, gA, A, expA},
  {μ, A} = μA;
  z = RandomReal[NormalDistribution[0, 1], {λ, dim}];
  expA = N@MatrixExp[A];
  x = (μ + expA.zᵀ)ᵀ;
  z = z[[Ordering[f /@ x]]];
  u = utilityFunction[λ];
  gμ = zᵀ.u;
  gA = Plus @@ MapThread[#1 (Outer[Times, #2, #2] - eye[dim]) &, {u, z}];
  {μ + ημ expA.gμ, A + (ηA / 2) gA}
 ]

learningRateXNES[d_] := N[3 (3 + Log[d]) / (5 d Sqrt[d])]

xNES[f_, dim_, μA_, λ_, ημ_, ηA_, nIter_] := Nest[xNESstep[f, dim, #, λ, ημ, ηA] &, μA, nIter]
```

Example Experiments

▪ Simple experiment

```
f := -Norm@# &
```

SNES

```
dim = 5;
λ = populationSize[dim];
η = learningRateSNES[dim];
μ = RandomReal[{-1, 1}, dim];
σ = Table[1., {dim}];
```

```
AbsoluteTiming@Round[sNES[f, dim, {μ, σ}, λ, η, η, 1000], 0.001]
```

```
{0.172666, {{0., 0., 0., 0., 0.}, {0., 0., 0., 0., 0.}}}
```

xNES

```
A = IdentityMatrix[dim];
```

```
ημ = 1; ηA = learningRateXNES[dim];
```

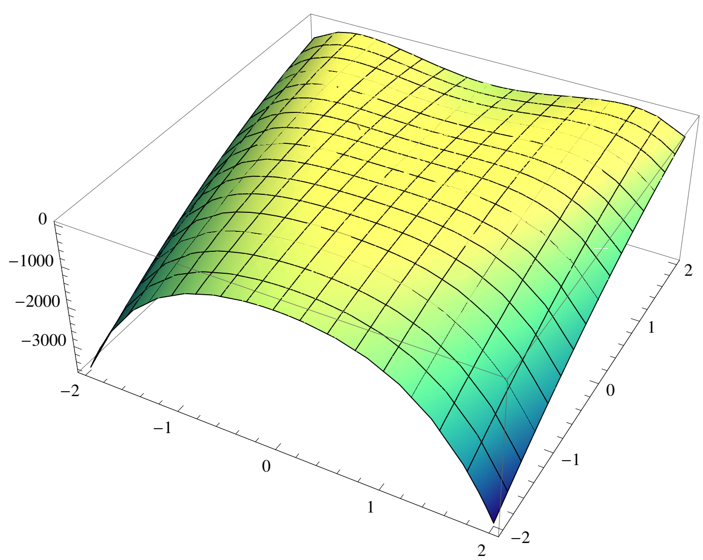
```
AbsoluteTiming@Round[xNES[f, dim, {μ, A}, λ, ημ, ηA, 1000], 0.001]
```

```
{0.392157, {{0., 0., 0., 0., 0.}, {{-49.313, -0.012, 0.067, -0.164, -0.224},
   {-0.012, -48.616, -0.105, -0.054, -0.03}, {0.067, -0.105, -48.529, 0.018, 0.256},
   {-0.164, -0.054, 0.018, -48.641, -0.229}, {-0.224, -0.03, 0.256, -0.229, -48.606}}}}
```

Rosenbrock Function

```
fitRosenbrock[x_] := -Total[100 (Most[x]^2 - Rest[x])^2 + (Most[x] - 1)^2]
```

```
Plot3D[fitRosenbrock[{x, y}], {x, -2, 2},
 {y, -2, 2}, PlotRange → All, ColorFunction → "BlueGreenYellow"]
```



SNES

```
dim = 3;
λ = populationSize[dim];
η = learningRateSNES[dim];
μ = RandomReal[{-1, 1}, dim];
σ = Table[1., {dim}];
```

```
AbsoluteTiming[sNES[fitRosenbrock, dim, {μ, σ}, λ, η, η, 50 000]]
```

$\{16.372856, \{\{0.999992, 0.999985, 0.99997\}, \{3.1341\times10^{-9}, 1.74963\times10^{-8}, 1.75847\times10^{-8}\}\}\}$

xNES

```
A = IdentityMatrix[dim];
ημ = 1; ηA = learningRateXNES[dim];
```

```
AbsoluteTiming[xNES[fitRosenbrock, dim, {μ, A}, λ, ημ, ηA, 5000]]
```

```
{2.624237, {{0.999159, 0.998328, 0.996656}, {{-42.0703, -3.21244, -0.981894},
   {-3.21244, -23.5909, 7.84494}, {-0.981894, 7.84494, -28.2068}}}}
```