

Kotlin ∇ : A Shape-safe DSL for Differentiable Programming

Breandan Considine, Michalis Famelis, Liam Paull

Main Idea

- We create an embedded DSL for differentiable programming in Kotlin
- Supports shape checking and inference for multi-dimensional arrays
- Implementable in any language with first-class functions and generics

Shape errors

There are three broad strategies for handling shape errors:

- Perform type coercion by implicitly broadcasting or reshaping arrays
- Raise a runtime error (e.g. `tf.errors.InvalidArgumentError`)
- Do not allow programs which can result in a shape error to compile

In Kotlin ∇ , we prefer the last strategy. Consider the following scenario:

```
a = np.array('0 1 2 3 4 5')
b = np.array('6 7 8 9')
c = a + b
```

```
val a = Vec(0, 1, 2, 3, 4, 5)
val b = Vec(6, 7, 8, 9)
val c = a + b
```

Similarly, when the inner dimensions of two matrices do not match:

```
a = np.matrix('0 1 2 ; 3 4 5')
b = np.matrix('6 7; 8 9')
c = a @ b
```

```
val a = Mat2x3(0, 1, 2, 3, 4, 5)
val b = Mat2x2(6, 7, 8, 9)
val c = a * b
```

We can detect the presence and location of the error within the editor.

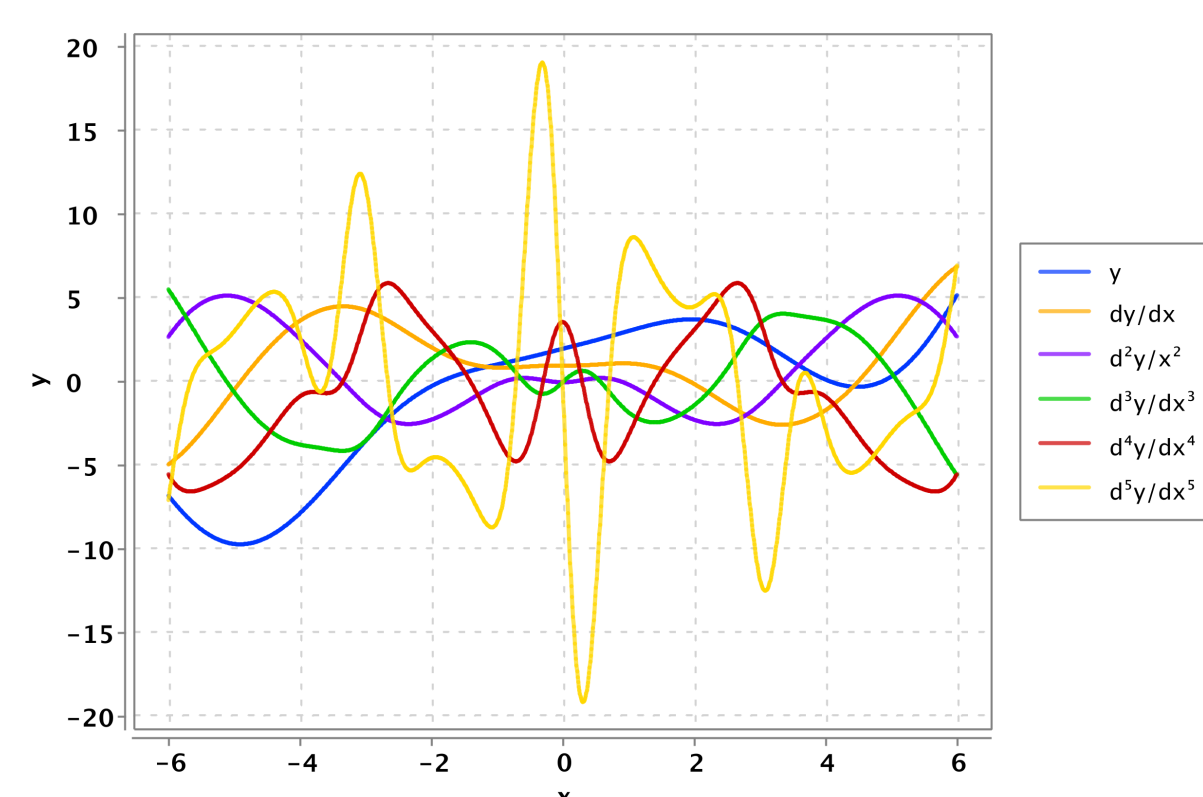
Type system

Math	Infix	Prefix	Postfix	Operator Type Signature
$A(B)$ $A \circ B$	<code>a(b)</code>			$(a: \mathbb{R}^r \rightarrow \mathbb{R}^p, b: \mathbb{R}^p \rightarrow \mathbb{R}^q) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R}^q)$
$A \pm B$	<code>a + b</code> <code>a - b</code>	<code>plus(a, b)</code> <code>minus(a, b)</code>		$(a: \mathbb{R}^r \rightarrow \mathbb{R}^p, b: \mathbb{R}^p \rightarrow \mathbb{R}^q) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R}^q)$
AB	<code>a * b</code> <code>a.times(b)</code>	<code>times(a, b)</code>		$(a: \mathbb{R}^r \rightarrow \mathbb{R}^{m \times n}, b: \mathbb{R}^n \rightarrow \mathbb{R}^{p \times q}) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R}^{m \times p})$
$\frac{A}{B}$ AB^{-1}	<code>a / b</code> <code>a.div(b)</code>	<code>div(a, b)</code>		$(a: \mathbb{R}^r \rightarrow \mathbb{R}^{m \times n}, b: \mathbb{R}^p \rightarrow \mathbb{R}^{p \times n}) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R}^{m \times p})$
$\log_b A$	<code>a.log(b)</code>	<code>log(a, b)</code>		$(a: \mathbb{R}^r \rightarrow \mathbb{R}^{m \times n}, b: \mathbb{R}^p \rightarrow \mathbb{R}^{p \times n}) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R})$
A^b	<code>a.pow(b)</code>	<code>pow(a, b)</code>		$(a: \mathbb{R}^r \rightarrow \mathbb{R}^{m \times n}, b: \mathbb{R}^p \rightarrow \mathbb{R}) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R}^{m \times n})$
∇a		<code>grad(a)</code>	<code>a.grad()</code>	$(a: C(\mathbb{R}^r \rightarrow \mathbb{R})) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R}^r)$
$\nabla_B a$	<code>a.d(b)</code> <code>a.grad(b)</code>	<code>grad(a, b)</code>		$(a: C(\mathbb{R}^r \rightarrow \mathbb{R}), b: C(\mathbb{R}^p \rightarrow \mathbb{R}^n)) \rightarrow (\mathbb{R}^p \rightarrow \mathbb{R}^n)$
$\nabla \cdot A$		<code>divg(a)</code>	<code>a.divg()</code>	$(a: C(\mathbb{R}^r \rightarrow \mathbb{R}^m)) \rightarrow (\mathbb{R}^r \rightarrow \mathbb{R})$
$\nabla \times A$		<code>curl(a)</code>	<code>a.curl()</code>	$(a: C(\mathbb{R}^3 \rightarrow \mathbb{R}^3)) \rightarrow (\mathbb{R}^3 \rightarrow \mathbb{R}^3)$

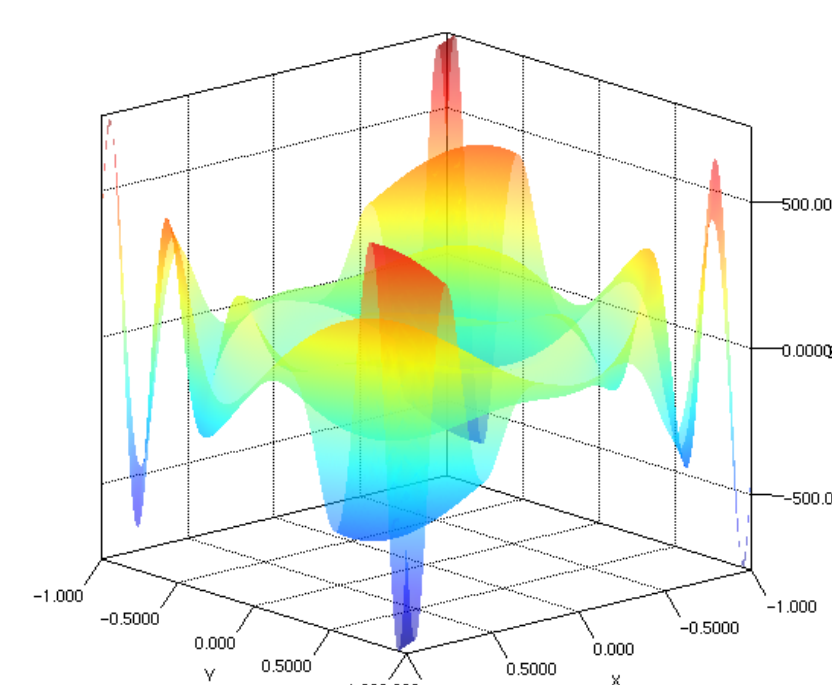
Visualization

Kotlin ∇ is capable of computing arbitrarily high order derivatives.

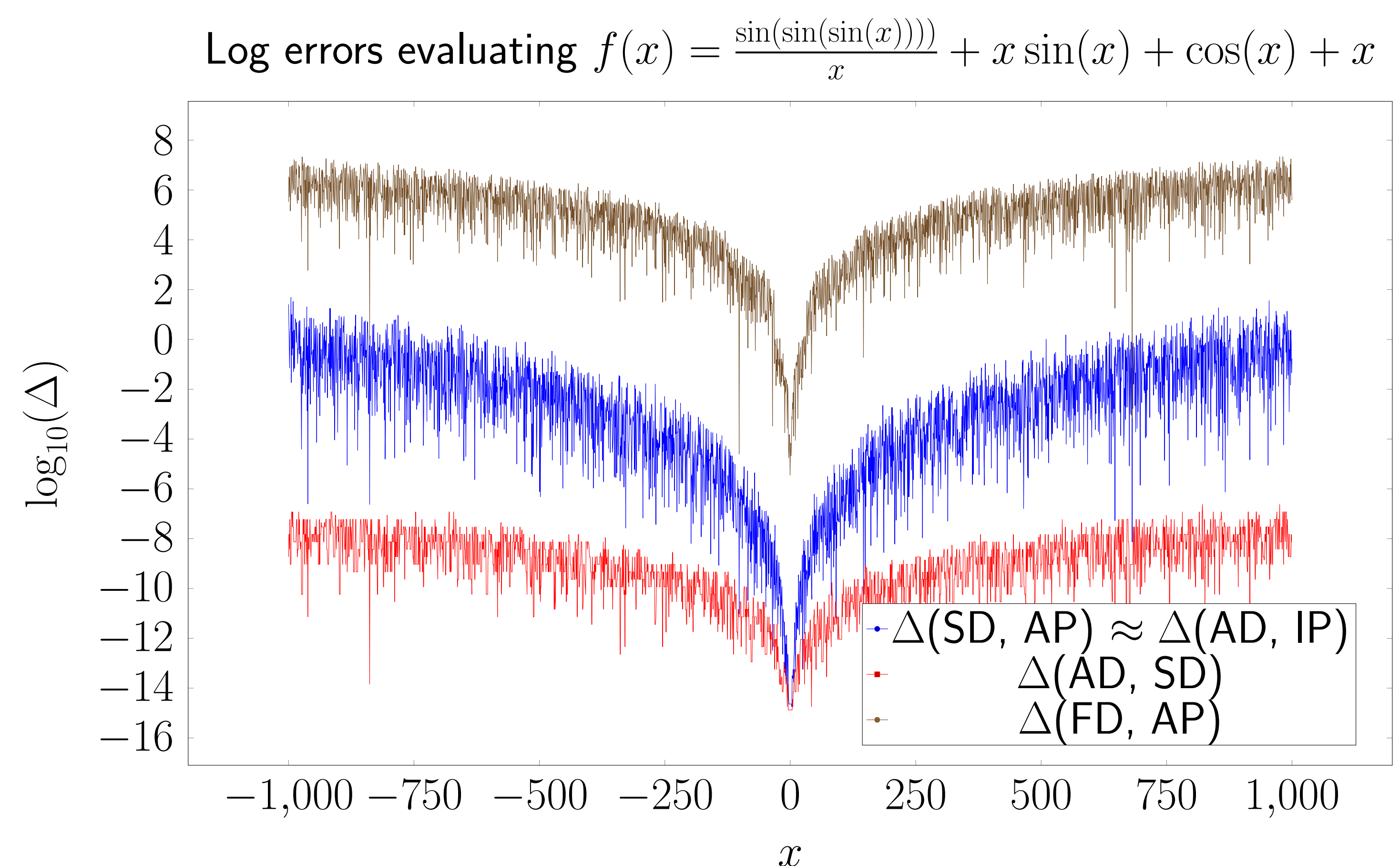
```
val y = sin(sin(sin(x))) / x
val z = y + sin(x) * x + cos(x) + x
val d1 = d(z) / d(x)
val d2 = d(d1) / d(x)
val d3 = d(d2) / d(x)
val d4 = d(d3) / d(x)
val d5 = d(d4) / d(x)
plot2D(-6..6, z, d1, d2, d3, d4, d5)
```



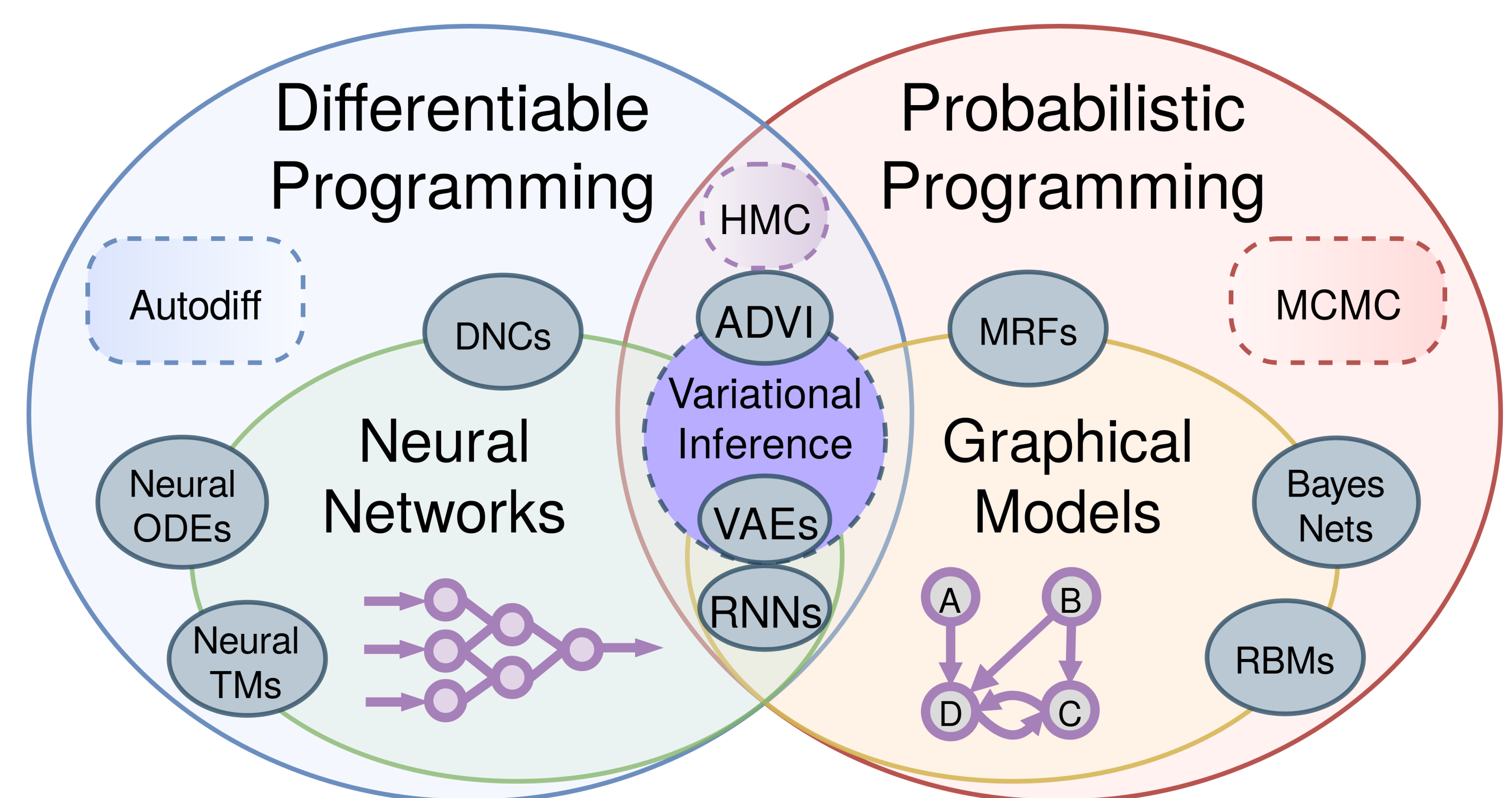
```
val Z = 10 * (x * x + pow(y, 2))
val sinZ = sin(Z)
val sinZ_10 = sinZ / 10
val dZ_dx = d(sinZ_10) / d(x)
val d2Z_dxdy = d(dZ_dx) / d(y)
val d3Z_d2xdy = grad(d2Z_dxdy)[x]
plot3D(-1..1, d3Z_d2xdy)
```



Testing

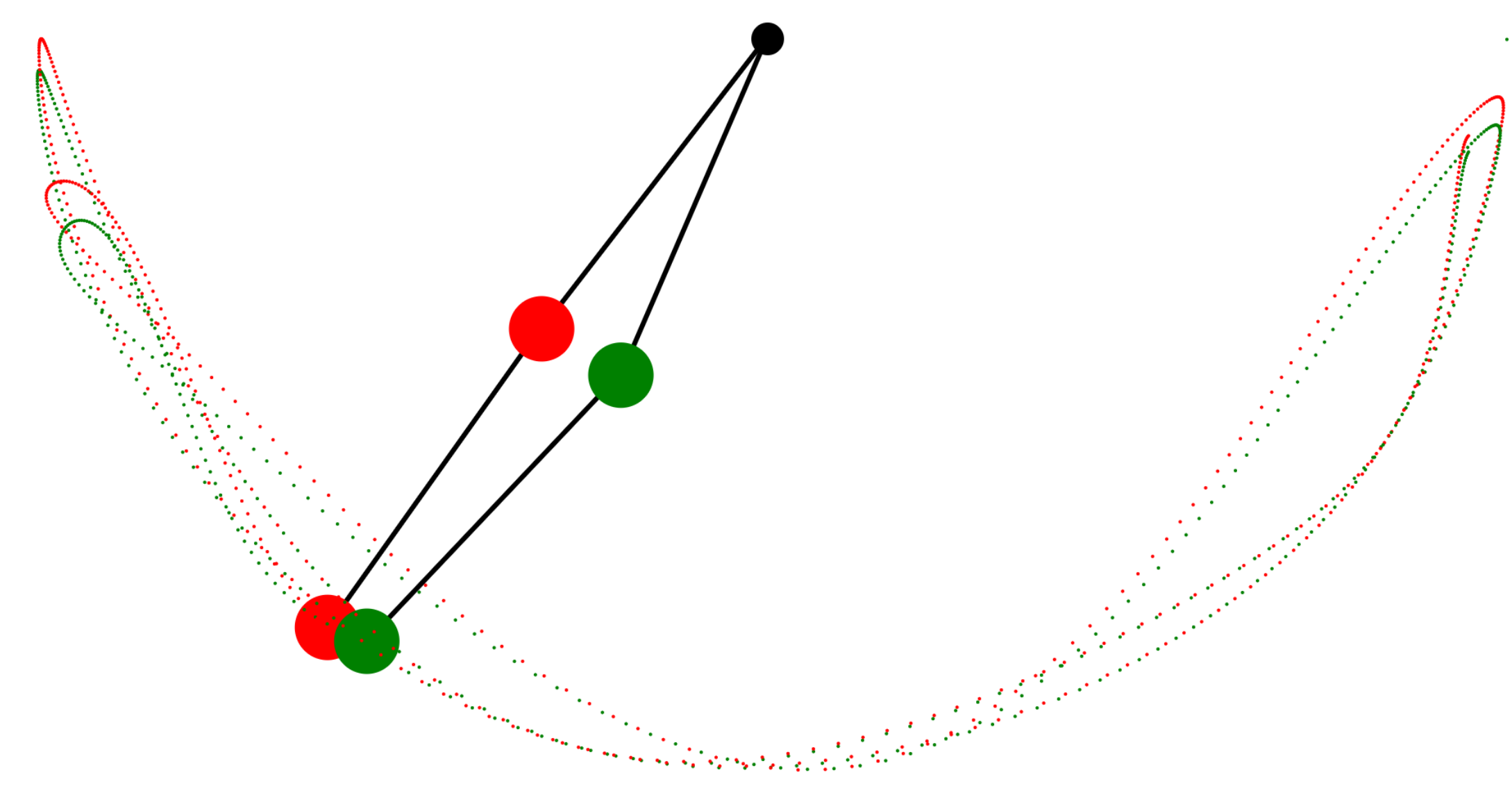


Differentiable Programming



Double Pendulum

$$\begin{aligned}\dot{\theta}_1 &= \frac{6}{m l^2} \frac{2 p_{\theta_1} - 3 \cos(\theta_1 - \theta_2) p_{\theta_2}}{16 - 9 \cos^2(\theta_1 - \theta_2)} \\ \dot{\theta}_2 &= \frac{6}{m l^2} \frac{8 p_{\theta_2} - 3 \cos(\theta_1 - \theta_2) p_{\theta_1}}{16 - 9 \cos^2(\theta_1 - \theta_2)} \\ \dot{p}_{\theta_1} &= \frac{\partial L}{\partial \theta_1} = -\frac{1}{2} m l^2 \left(\dot{\theta}_1 \dot{\theta}_2 \sin(\theta_1 - \theta_2) + 3 \frac{g}{l} \sin \theta_1 \right) \\ \dot{p}_{\theta_2} &= \frac{\partial L}{\partial \theta_2} = -\frac{1}{2} m l^2 \left(-\dot{\theta}_1 \dot{\theta}_2 \sin(\theta_1 - \theta_2) + \frac{g}{l} \sin \theta_2 \right).\end{aligned}$$



Université
de Montréal

