Accelerate / vDSP / Vector generation

API Collection

Vector generation

Populate vectors with ramps, values from lookup tables, interpolated values, and window functions.

Topics

Vector generation with ramps using an initial value and increment

```
static func ramp(withInitialValue: Float, increment: Float, count: Int)
-> [Float]
```

Returns a single-precision vector that contains monotonically incrementing or decrementing values using an initial value and increment.

```
static func ramp(withInitialValue: Double, increment: Double, count:
Int) -> [Double]
```

Returns a double-precision vector that contains monotonically incrementing or decrementing values using an initial value and increment.

```
static func formRamp<V>(withInitialValue: Float, increment: Float,
result: inout V)
```

Populates a single-precision vector with monotonically incrementing or decrementing values using an initial value and increment.

```
static func formRamp<V>(withInitialValue: Double, increment: Double,
result: inout V)
```

Populates a double-precision vector with monotonically incrementing or decrementing values using an initial value and increment.

```
vDSP_vramp
```

Generates a single-precision vector with monotonically incrementing or decrementing values using an initial value and increment.

```
vDSP_vrampD
```

Generates a double-precision vector with monotonically incrementing or decrementing values using an initial value and increment.

Vector generation with ramps using a range

```
static func ramp(in: ClosedRange<Float>, count: Int) -> [Float]
```

Returns a double-precision vector that contains monotonically incrementing or decrementing values within a range.

```
static func ramp(in: ClosedRange<Double>, count: Int) -> [Double]
```

Returns a single-precision vector that contains monotonically incrementing or decrementing values within a range.

```
static func formRamp<V>(in: ClosedRange<Float>, result: inout V)
```

Populates a double-precision vector with monotonically incrementing or decrementing values within a range.

```
static func formRamp<V>(in: ClosedRange<Double>, result: inout V)
```

Populates a single-precision vector with monotonically incrementing or decrementing values within a range.

```
vDSP_vgen
```

Generates a single-precision vector that contains monotonically incrementing or decrementing values within a range.

```
vDSP_vgenD
```

Generates a double-precision vector that contains monotonically incrementing or decrementing values within a range.

Vector generation with ramps and multiplication by a second vector

```
static func ramp<U>(withInitialValue: inout Float, multiplyingBy: U,
increment: Float) -> [Float]
```

Returns a single-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`static func ramp<U>(withInitialValue: inout Double, multiplyingBy: U, increment: Double) -> [Double]`

Returns a double-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`static func formRamp<U, V>(withInitialValue: inout Float, multiplyingBy: U, increment: Float, result: inout V)`

Populates a single-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`static func formRamp<U, V>(withInitialValue: inout Double, multiplyingBy: U, increment: Double, result: inout V)`

Populates a double-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmul`

Generates a single-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmulD`

Generates a double-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmul_s1_15`

Generates a fixed-point 1.15 format vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmul_s8_24`

Generates a fixed-point 8.24 format vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

Vector addition with ramps and multiplication by a second vector

`vDSP_vrampmuladd`

Adds a single-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmuladdD`

Adds a double-precision vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmuladd_s1_15`

Adds a fixed-point 1.15 format vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

`vDSP_vrampmuladd_s8_24`

Adds a fixed-point 8.24 format vector that contains monotonically incrementing or decrementing values, and multiplies that vector by a source vector.

Vector generation by extrapolation and interpolation

`static func linearInterpolate<T, U>(values: T, atIndices: U) -> [Float]`

Returns the single-precision linearly interpolated values of a vector at the specified indices.

`static func linearInterpolate<T, U>(values: T, atIndices: U) -> [Double]`

Returns the double-precision linearly interpolated values of a vector at the specified indices.

`static func linearInterpolate<T, U, V>(values: T, atIndices: U, result: inout V)`

Computes the double-precision linearly interpolated values of a vector at the specified indices.

`static func linearInterpolate<T, U, V>(values: T, atIndices: U, result: inout V)`

Computes the single-precision linearly interpolated values of a vector at the specified indices.

`vDSP_vgenp`

Generates the single-precision linearly interpolated values of a vector at the specified indices.

`vDSP_vgenpD`

Generates the double-precision linearly interpolated values of a vector at the specified indices.

Vector generation with lookup tables

`static func linearInterpolate<T, U>(lookupTable: T, withOffsets: U, scale: Double, baseOffset: Double) -> [Double]`

Returns the double-precision linearly interpolated values of a lookup table from the specified offsets.

`static func linearInterpolate<T, U>(lookupTable: T, withOffsets: U, scale: Float, baseOffset: Float) -> [Float]`

Returns the single-precision linearly interpolated values of a lookup table from the specified offsets.

`static func linearInterpolate<T, U, V>(lookupTable: T, withOffsets: U, scale: Double, baseOffset: Double, result: inout V)`

Computes the double-precision linearly interpolated values of a lookup table from the specified offsets.

`static func linearInterpolate<T, U, V>(lookupTable: T, withOffsets: U, scale: Float, baseOffset: Float, result: inout V)`

Computes the single-precision linearly interpolated values of a lookup table from the specified offsets.

`vDSP_vtabi`

Generates a single-precision vector by interpolating values from a lookup table.

`vDSP_vtabiD`

Generates a double-precision vector by interpolating values from a lookup table.

Vector generation with window functions

Reducing spectral leakage with windowing

Multiply signal data by window sequence values when performing transforms with noninteger period signals.

`static func window<T>(ofType: T.Type, usingSequence: vDSP.WindowSequence, count: Int, isHalfWindow: Bool) -> [T]`

Returns an array that contains the specified window.

`static func formWindow<V>(usingSequence: vDSP.WindowSequence, result: inout V, isHalfWindow: Bool)`

Populates a double-precision vector with a specified window.

`static func formWindow<V>(usingSequence: vDSP.WindowSequence, result: inout V, isHalfWindow: Bool)`

Populates a single-precision vector with a specified window.

`enum WindowSequence`

Constants that specify window sequence functions.

`vDSP_blkman_window`

Creates a single-precision Blackman window.

`vDSP_blkman_windowD`

Creates a double-precision Blackman window.

`vDSP_hamm_window`

Creates a single-precision Hamming window.

`vDSP_hamm_windowD`

Creates a double-precision Hamming window.

`vDSP_hann_window`

Creates a single-precision Hann window.

`vDSP_hann_windowD`

Creates a double-precision Hann window.

`var vDSP_HALF_WINDOW: Int`

Specifies that the window should only contain the bottom half of the values (`0` to `(N+1)/2`).

`var vDSP_HANN_DENORM: Int`

Specifies a denormalized Hann window.

`var vDSP_HANN_NORM: Int`

Specifies a normalized Hann window

Stereo ramp generation

`static func stereoRamp<U>(withInitialValue: inout Double, multiplyingBy: U, U, increment: Double) -> (firstOutput: [Double], secondOutput: [Double])`

Returns two double-precision vectors that contain stereo monotonically incrementing or decrementing values multiplied by two source vectors.

`static func stereoRamp<U>(withInitialValue: inout Float, multiplyingBy: U, U, increment: Float) -> (firstOutput: [Float], secondOutput: [Float])`

Returns two single-precision vectors that contain stereo monotonically incrementing or decrementing values multiplied by two source vectors.

```
static func formStereoRamp<U, V>(withInitialValue: inout Double,
multiplyingBy: U, U, increment: Double, results: inout V, inout V)
```

Populates two single-precision vectors that contain stereo monotonically incrementing or decrementing values multiplied by two source vectors.

```
static func formStereoRamp<U, V>(withInitialValue: inout Float,
multiplyingBy: U, U, increment: Float, results: inout V, inout V)
```

Populates two single-precision vectors that contain stereo monotonically incrementing or decrementing values multiplied by two source vectors.

`vDSP_vrampmul2`

Generates a single-precision, stereo ramped vector and multiplies that vector by an input vector.

`vDSP_vrampmul2D`

Generates a double-precision, stereo ramped vector and multiplies that vector by an input vector.

`vDSP_vrampmul2_s1_15`

Generates a fixed-point, 1.15 format, stereo ramped vector and multiplies that vector by an input vector.

`vDSP_vrampmul2_s8_24`

Generates a fixed-point, 8.24 format, stereo ramped vector and multiplies that vector by an input vector.

`vDSP_vrampmuladd2`

Multiplies a single-precision, stereo input vector by a value that ramps up on successive calls, and cumulatively adds the result to the output vector.

`vDSP_vrampmuladd2D`

Multiplies a double-precision, stereo input vector by a value that ramps up on successive calls, and cumulatively adds the result to the output vector.

`vDSP_vrampmuladd2_s1_15`

Multiplies a fixed-point, 1.15 format, stereo input vector by a value that ramps on successive calls, and adds the result to the output vector.

`vDSP_vrampmuladd2_s8_24`

Multiplies a fixed-point, 8.24 format, stereo input vector by a value that ramps on successive calls, and adds the result to the output vector.

See Also

Vector generation, filling, and clearing

Vector clear and fill functions

Populate vectors with zeros or a scalar value.