The Sigmoid Function and the Universal Approximation Theorem

Dustin Bryant, Jim Woodcock, and Simon Foster

September 1, 2025

Abstract

We present a machine-checked Isabelle/HOL development of the sigmoid function

$$\sigma(x) = \frac{e^x}{1 + e^x},$$

together with its most important analytic properties. After proving positivity, strict monotonicity, C^∞ smoothness, and the limits at $\pm\infty$, we derive a closed-form expression for the n–th derivative using Stirling numbers of the second kind, following the combinatorial argument of Minai and Williams [4]. These results are packaged into a small reusable library of lemmas on σ.

Building on this analytic groundwork we mechanise a constructive version of the classical Universal Approximation Theorem: for every continuous function $f : [a, b] \to \mathbb{R}$ and every $\varepsilon > 0$ there is a single-hidden-layer neural network with sigmoidal activations whose output is within ε of f everywhere on $[a, b]$. Our proof follows the method of Costarell and Spigler [2], giving the first fully verified end-to-end proof of this theorem inside a higher-order proof assistant.

Contents

1 Limits and Higher Order Derivatives **2**
1.1 ε–δ Characterizations of Limits and Continuity 2
1.2 Nth Order Derivatives and $C^k(U)$ Smoothness 7

2 Definition and Analytical Properties **8**
2.1 Range, Monotonicity, and Symmetry 9
2.2 Differentiability and Derivative Identities 10
2.3 Logit, Softmax, and the Tanh Connection 11

3 Derivative Identities and Smoothness **12**

4 Asymptotic and Qualitative Properties **19**
4.1 Limits at Infinity of Sigmoid and its Derivative 19
4.2 Curvature and Inflection 20
4.3 Monotonicity and Bounds of the First Derivative 21
4.4 Sigmoidal and Heaviside Step Functions 23
4.5 Uniform Approximation by Sigmoids 23

5 Universal Approximation Theorem **24**

1 Limits and Higher Order Derivatives

theory *Limits-Higher-Order-Derivatives*
imports *HOL−Analysis.Analysis*
begin

1.1 ε–δ Characterizations of Limits and Continuity

lemma *tendsto-at-top-epsilon-def*:
$(f \longrightarrow L)$ *at-top* $= (\forall\ \varepsilon > 0.\ \exists N.\ \forall x \geq N.\ |(f\ (x::real)::real) - L| < \varepsilon)$
by (*simp add*: *Zfun-def tendsto-Zfun-iff eventually-at-top-linorder*)

lemma *tendsto-at-bot-epsilon-def*:
$(f \longrightarrow L)$ *at-bot* $= (\forall\ \varepsilon > 0.\ \exists N.\ \forall x \leq N.\ |(f\ (x::real)::real) - L| < \varepsilon)$
by (*simp add*: *Zfun-def tendsto-Zfun-iff eventually-at-bot-linorder*)

lemma *tendsto-inf-at-top-epsilon-def*:
$(g \longrightarrow \infty)$ *at-top* $= (\forall\ \varepsilon > 0.\ \exists N.\ \forall x \geq N.\ (g\ (x::real)::real) > \varepsilon)$
by (*subst tendsto-PInfty′*, *subst Filter.eventually-at-top-linorder*, *simp*)

lemma *tendsto-inf-at-bot-epsilon-def*:
$(g \longrightarrow \infty)$ *at-bot* $= (\forall\ \varepsilon > 0.\ \exists N.\ \forall x \leq N.\ (g\ (x::real)::real) > \varepsilon)$
by (*subst tendsto-PInfty′*, *subst Filter.eventually-at-bot-linorder*, *simp*)

lemma *tendsto-minus-inf-at-top-epsilon-def*:
$(g \longrightarrow -\infty)$ *at-top* $= (\forall\ \varepsilon < 0.\ \exists N.\ \forall x \geq N.\ (g\ (x::real)::real) < \varepsilon)$
by(*subst tendsto-MInfty′*, *subst Filter.eventually-at-top-linorder*, *simp*)

lemma *tendsto-minus-inf-at-bot-epsilon-def*:
$(g \longrightarrow -\infty)$ *at-bot* $= (\forall\ \varepsilon < 0.\ \exists N.\ \forall x \leq N.\ (g\ (x::real)::real) < \varepsilon)$
by (*subst tendsto-MInfty′*, *subst Filter.eventually-at-bot-linorder*, *simp*)

lemma *tendsto-at-x-epsilon-def*:
fixes $f :: real \Rightarrow real$ **and** $L :: real$ **and** $x :: real$
shows $(f \longrightarrow L)\ (at\ x) = (\forall \varepsilon > 0.\ \exists \delta > 0.\ \forall y.\ (y \neq x \wedge |y - x| < \delta) \longrightarrow |f\ y - L| < \varepsilon)$
unfolding *tendsto-def*
proof (*subst eventually-at*, *safe*)

— First Direction — We show that the filter definition implies the ε–δ formulation.

fix ε :: *real*
assume *lim-neigh*: $\forall S.\ open\ S \longrightarrow L \in S \longrightarrow (\exists d{>}0.\ \forall xa{\in}UNIV.\ xa \neq x \wedge dist\ xa\ x < d \longrightarrow f\ xa \in S)$
assume *ε-pos*: $0 < \varepsilon$
show $\exists \delta{>}0.\ \forall y.\ y \neq x \wedge |y - x| < \delta \longrightarrow |f\ y - L| < \varepsilon$
proof –

Choose S as the open ball around L with radius ε.

have *open* (*ball L* ε)
by *simp*

Confirm that L lies in the ball.

moreover have $L \in ball\ L\ \varepsilon$
unfolding *ball-def* **by** (*simp add*: *ε-pos*)

By applying lim_neigh to the ball, we obtain a suitable δ.

ultimately obtain δ **where** *d-pos*: $\delta > 0$
and *δ-prop*: $\forall y.\ y \neq x \wedge dist\ y\ x < \delta \longrightarrow f\ y \in ball\ L\ \varepsilon$
by (*meson UNIV-I lim-neigh*)

Since $f(y) \in \text{ball}(L, \varepsilon)$ means $|f(y) - L| < \varepsilon$, we deduce the $\varepsilon\delta$ condition.

hence $\forall y.\ y \neq x \wedge |y - x| < \delta \longrightarrow |f\ y - L| < \varepsilon$
by (*auto simp*: *ball-def dist-norm*)
thus *?thesis*
using *d-pos* **by** *blast*
qed
next

— Second Direction — We show that the ε–δ formulation implies the filter definition.

fix S :: *real set*
assume *eps-delta*: $\forall \varepsilon{>}0.\ \exists \delta{>}0.\ \forall y.\ (y \neq x \wedge |y - x| < \delta) \longrightarrow |f\ y - L| < \varepsilon$
and *S-open*: *open S*
and *L-in-S*: $L \in S$

Since S is open and contains L, there exists an ε-ball around L contained in S.

from *S-open L-in-S* **obtain** ε **where** *ε-pos*: $\varepsilon > 0$ **and** *ball-sub*: $ball\ L\ \varepsilon \subseteq S$
by (*meson openE*)

Applying the ε–δ assumption for this particular ε yields a $\delta > 0$ such that for all y, if $y \neq x$ and $|y - x| < \delta$ then $|f(y) - L| < \varepsilon$.

from *eps-delta* **obtain** δ **where** *δ-pos*: $\delta > 0$
and *δ-prop*: $\forall y.\ (y \neq x \wedge |y - x| < \delta) \longrightarrow |f\ y - L| < \varepsilon$
using *ε-pos* **by** *blast*

Notice that $|f(y) - L| < \varepsilon$ is equivalent to $f(y) \in \text{ball}\, L\, \varepsilon$.

have $\forall y.\ (y \neq x \wedge \textit{dist}\ y\ x < \delta) \longrightarrow f\ y \in \textit{ball}\ L\ \varepsilon$
using *δ-prop dist-real-def* **by** *fastforce*

Since $\text{ball}(L, \varepsilon) \subseteq S$, for all y with $y \neq x$ and $\text{dist}\, y\, x < \delta$, we have $f\, y \in S$.

hence $\forall y.\ (y \neq x \wedge \textit{dist}\ y\ x < \delta) \longrightarrow f\ y \in S$
using *ball-sub* **by** *blast*

This gives exactly the existence of some d (namely δ) satisfying the filter condition.

thus $\exists d{>}0.\ \forall y{\in}\textit{UNIV}.\ (y \neq x \wedge \textit{dist}\ y\ x < d) \longrightarrow f\ y \in S$
using *δ-pos* **by** *blast*
qed

lemma *continuous-at-eps-delta*:
fixes $g :: \textit{real} \Rightarrow \textit{real}$ **and** $y :: \textit{real}$
shows $\textit{continuous}\ (\textit{at}\ y)\ g = (\forall \varepsilon > 0.\ \exists \delta > 0.\ \forall x.\ |x - y| < \delta \longrightarrow |g\ x - g\ y| < \varepsilon)$
proof −
have $\textit{continuous}\ (\textit{at}\ y)\ g = (\forall \varepsilon > 0.\ \exists \delta > 0.\ \forall x.\ (x \neq y \wedge |x - y| < \delta) \longrightarrow |g\ x - g\ y| < \varepsilon)$
by (*simp add*: *isCont-def tendsto-at-x-epsilon-def*)
also have ... $= (\forall \varepsilon > 0.\ \exists \delta > 0.\ \forall x.\ |x - y| < \delta \longrightarrow |g\ x - g\ y| < \varepsilon)$
by (*metis abs-eq-0 diff-self*)
finally show *?thesis*.
qed

lemma *tendsto-divide-approaches-const*:
fixes $f\ g :: \textit{real} \Rightarrow \textit{real}$
assumes *f-lim*: $((\lambda x.\ f\ (x{::}\textit{real})) \longrightarrow c)\ \textit{at-top}$
and *g-lim*: $((\lambda x.\ g\ (x{::}\textit{real})) \longrightarrow \infty)\ \textit{at-top}$
shows $((\lambda x.\ f\ (x{::}\textit{real})\ /\ g\ x) \longrightarrow 0)\ \textit{at-top}$
proof(*subst tendsto-at-top-epsilon-def*, *clarify*)
fix $\varepsilon :: \textit{real}$
assume *ε-pos*: $0 < \varepsilon$

obtain M **where** *M-def*: $M = \textit{abs}\ c + 1$ **and** *M-gt-0*: $M > 0$
by *simp*

obtain $N1$ **where** *N1-def*: $\forall x{\geq}N1.\ \textit{abs}\ (f\ x - c) < 1$
using *f-lim tendsto-at-top-epsilon-def zero-less-one* **by** *blast*

have *f-bound*: $\forall x{\geq}N1.\ \textit{abs}\ (f\ x) < M$
using *M-def N1-def* **by** *fastforce*

have *M-over-ε-gt-0*: $M\ /\ \varepsilon > 0$
by (*simp add*: *M-gt-0 ε-pos*)

then obtain $N2$ **where** *N2-def*: $\forall x{\geq}N2.\ g\ x > M\ /\ \varepsilon$

using *g-lim tendsto-inf-at-top-epsilon-def* **by** *blast*

obtain *N* **where** *N* = *max N1 N2* **and** *N-ge-N1*: *N* ≥ *N1* **and** *N-ge-N2*: *N* ≥ *N2*
by *auto*

show ∃ *N*::*real*. ∀ *x*≥*N*. |*f x* / *g x* − *0*| < *ε*
proof(*intro exI* [**where** *x*=*N*], *clarify*)
fix *x* :: *real*
assume *x-ge-N*: *N* ≤ *x*

have *f-bound-x*: |*f x*| < *M*
using *N-ge-N1 f-bound x-ge-N* **by** *auto*

have *g-bound-x*: *g x* > *M* / *ε*
using *N2-def N-ge-N2 x-ge-N* **by** *auto*

have |*f x* / *g x*| = |*f x*| / |*g x*|
using *abs-divide* **by** *blast*
also have ... < *M* / |*g x*|
using *M-over-ε-gt-0 divide-strict-right-mono f-bound-x g-bound-x* **by** *force*
also have ... < *ε*
by (*metis M-over-ε-gt-0 ε-pos abs-real-def g-bound-x mult.commute order-less-irrefl order-less-trans pos-divide-less-eq*)
finally show |*f x* / *g x* − *0*| < *ε*
by *linarith*
qed
qed

lemma *tendsto-divide-approaches-const-at-bot*:
fixes *f g* :: *real* ⇒ *real*
assumes *f-lim*: ((λ*x*. *f* (*x*::*real*)) ⟶ *c*) *at-bot*
and *g-lim*: ((λ*x*. *g* (*x*::*real*)) ⟶ ∞) *at-bot*
shows ((λ*x*. *f* (*x*::*real*) / *g x*) ⟶ *0*) *at-bot*
proof(*subst tendsto-at-bot-epsilon-def*, *clarify*)
fix *ε* :: *real*
assume *ε-pos*: *0* < *ε*

obtain *M* **where** *M-def*: *M* = *abs c* + *1* **and** *M-gt-0*: *M* > *0*
by *simp*

obtain *N1* **where** *N1-def*: ∀ *x*≤*N1*. *abs* (*f x* − *c*) < *1*
using *f-lim tendsto-at-bot-epsilon-def zero-less-one* **by** *blast*

have *f-bound*: ∀ *x*≤*N1*. *abs* (*f x*) < *M*
using *M-def N1-def* **by** *fastforce*

have *M-over-ε-gt-0*: *M* / *ε* > *0*
by (*simp add*: *M-gt-0 ε-pos*)

then obtain *N2* **where** *N2-def*: ∀ *x*≤*N2*. *g x* > *M* / *ε*
using *g-lim tendsto-inf-at-bot-epsilon-def* **by** *blast*

obtain *N* **where** *N* = *min N1 N2* **and** *N-le-N1*: *N* ≤ *N1* **and** *N-le-N2*: *N* ≤ *N2*
by *auto*

show ∃ *N*::*real*. ∀ *x*≤*N*. |*f x* / *g x* − *0*| < *ε*
proof(*intro exI* [**where** *x*=*N*], *clarify*)
fix *x* :: *real*
assume *x-le-N*: *x* ≤ *N*

have *f-bound-x*: |*f x*| < *M*
using *N-le-N1 f-bound x-le-N* **by** *auto*

have *g-bound-x*: *g x* > *M* / *ε*
using *N2-def N-le-N2 x-le-N* **by** *auto*

have |*f x* / *g x*| = |*f x*| / |*g x*|
using *abs-divide* **by** *blast*
also have ... < *M* / |*g x*|
using *M-over-ε-gt-0 divide-strict-right-mono f-bound-x g-bound-x* **by** *force*
also have ... < *ε*
by (*metis M-over-ε-gt-0 ε-pos abs-real-def g-bound-x mult.commute order-less-irrefl order-less-trans pos-divide-less-eq*)
finally show |*f x* / *g x* − *0*| < *ε*
by *linarith*
qed
qed

lemma *equal-limits-diff-zero-at-top*:
assumes *f-lim*: (*f* ⟶ (*L1*::*real*)) *at-top*
assumes *g-lim*: (*g* ⟶ (*L2*::*real*)) *at-top*
shows ((*f* − *g*) ⟶ (*L1* − *L2*)) *at-top*
proof −
have ((*λx*. *f x* − *g x*) ⟶ *L1* − *L2*) *at-top*
by (*rule tendsto-diff*, *rule f-lim*, *rule g-lim*)
then show *?thesis*
by (*simp add*: *fun-diff-def*)
qed

lemma *equal-limits-diff-zero-at-bot*:
assumes *f-lim*: (*f* ⟶ (*L1*::*real*)) *at-bot*
assumes *g-lim*: (*g* ⟶ (*L2*::*real*)) *at-bot*
shows ((*f* − *g*) ⟶ (*L1* − *L2*)) *at-bot*
proof −
have ((*λx*. *f x* − *g x*) ⟶ *L1* − *L2*) *at-bot*
by (*rule tendsto-diff*, *rule f-lim*, *rule g-lim*)

then show *?thesis*
by (*simp add*: *fun-diff-def*)
qed

1.2 Nth Order Derivatives and $C^k(U)$ Smoothness

fun *Nth-derivative* :: *nat* ⇒ (*real* ⇒ *real*) ⇒ (*real* ⇒ *real*) **where**
Nth-derivative 0 f = *f* |
Nth-derivative (*Suc n*) *f* = *deriv* (*Nth-derivative n f*)

lemma *first-derivative-alt-def*:
Nth-derivative 1 f = *deriv f*
by *simp*

lemma *second-derivative-alt-def*:
Nth-derivative 2 f = *deriv* (*deriv f*)
by (*simp add*: *numeral-2-eq-2*)

lemma *limit-def-nth-deriv*:
fixes *f* :: *real* ⇒ *real* **and** *a* :: *real* **and** *n* :: *nat*
assumes *n-pos*: *n* > *0*
and *D-last*: *DERIV* (*Nth-derivative* (*n* − *1*) *f*) *a* :> *Nth-derivative n f a*
shows
((*λx*. (*Nth-derivative* (*n* − *1*) *f x* − *Nth-derivative* (*n* − *1*) *f a*) / (*x* − *a*))
⟶ *Nth-derivative n f a*) (*at a*)
using *D-last has-field-derivativeD* **by** *blast*

definition *C-k-on* :: *nat* ⇒ (*real* ⇒ *real*) ⇒ *real set* ⇒ *bool* **where**
C-k-on k f U ≡
(*if k* = *0 then* (*open U* ∧ *continuous-on U f*)
else (*open U* ∧ (∀ *n* < *k*. (*Nth-derivative n f*) *differentiable-on U*
∧ *continuous-on U* (*Nth-derivative* (*Suc n*) *f*))))

lemma *C0-on-def*:
C-k-on 0 f U ⟷ (*open U* ∧ *continuous-on U f*)
by (*simp add*: *C-k-on-def*)

lemma *C1-cont-diff*:
assumes *C-k-on 1 f U*
shows *f differentiable-on U* ∧ *continuous-on U* (*deriv f*) ∧
(∀ *y* ∈ *U*. (*f has-real-derivative* (*deriv f*) *y*) (*at y*))
using *C-k-on-def DERIV-deriv-iff-real-differentiable assms at-within-open differentiable-on-def* **by** *fastforce*

lemma *C2-cont-diff*:
fixes *f* :: *real* ⇒ *real* **and** *U* :: *real set*
assumes *C-k-on 2 f U*
shows *f differentiable-on U* ∧ *continuous-on U* (*deriv f*) ∧
(∀ *y* ∈ *U*. (*f has-real-derivative* (*deriv f*) *y*) (*at y*)) ∧

deriv f differentiable-on U ∧ *continuous-on U* (*deriv* (*deriv f*)) ∧
(∀ *y* ∈ *U*. (*deriv f has-real-derivative* (*deriv* (*deriv f*)) *y*) (*at y*))
by (*smt* (*verit*, *best*) *C1-cont-diff C-k-on-def Nth-derivative.simps*(*1*,*2*) *One-nat-def assms less-2-cases-iff less-numeral-extra*(*1*) *nat-1-add-1 order.asym pos-add-strict*)

lemma *C2-on-open-U-def2*:
fixes *f* :: *real* ⇒ *real*
assumes *openU* : *open U*
and *diff-f* : *f differentiable-on U*
and *diff-df*: *deriv f differentiable-on U*
and *cont-d2f*: *continuous-on U* (*deriv* (*deriv f*))
shows *C-k-on 2 f U*
by (*simp add*: *C-k-on-def cont-d2f diff-df diff-f differentiable-imp-continuous-on less-2-cases-iff openU*)

lemma *C-k-on-subset*:
assumes *C-k-on k f U*
assumes *open-subset*: *open S* ∧ *S* ⊂ *U*
shows *C-k-on k f S*
using *assms*
by (*smt* (*verit*) *C-k-on-def continuous-on-subset differentiable-on-eq-differentiable-at dual-order.strict-implies-order subset-eq*)

definition *smooth-on* :: (*real* ⇒ *real*) ⇒ *real set* ⇒ *bool* **where**
smooth-on f U ≡ ∀ *k*. *C-k-on k f U*

end
theory *Sigmoid-Definition*
imports *HOL−Analysis.Analysis HOL−Combinatorics.Stirling Limits-Higher-Order-Derivatives*
begin

2 Definition and Analytical Properties

definition *sigmoid* :: *real* ⇒ *real* **where**
sigmoid x = *exp x* / (*1* + *exp x*)

lemma *sigmoid-alt-def*: *sigmoid x* = *inverse* (*1* + *exp*(−*x*))
proof −
have *sigmoid x* = (*exp*(*x*) ∗ *exp*(−*x*)) / ((*1* + *exp*(*x*))∗ *exp*(−*x*))
unfolding *sigmoid-def* **by** *simp*
also have ... = *1* / (*1*∗*exp*(−*x*) + *exp*(*x*)∗*exp*(−*x*))
by (*simp add*: *distrib-right exp-minus-inverse*)
also have ... = *inverse* (*exp*(−*x*) + *1*)
by (*simp add*: *divide-inverse-commute exp-minus*)
finally show *?thesis*
by *simp*
qed

2.1 Range, Monotonicity, and Symmetry

Bounds

lemma *sigmoid-pos*: *sigmoid x > 0*
by (*smt* (*verit*) *divide-le-0-1-iff exp-gt-zero inverse-eq-divide sigmoid-alt-def*)

Prove that $\sigma(x) < 1$ for all x.

lemma *sigmoid-less-1*: *sigmoid x < 1*
by (*smt* (*verit*) *le-divide-eq-1-pos not-exp-le-zero sigmoid-def*)

The sigmoid function $\sigma(x)$ satisfies

$$0 < \sigma(x) < 1 \quad \text{for all } x \in \mathbb{R}.$$

corollary *sigmoid-range*: *0 < sigmoid x ∧ sigmoid x < 1*
by (*simp add*: *sigmoid-less-1 sigmoid-pos*)

Symmetry around the origin: The sigmoid function σ satisfies

$$\sigma(-x) = 1 - \sigma(x) \quad \text{for all } x \in \mathbb{R},$$

reflecting that negative inputs shift the output towards 0, while positive inputs shift it towards 1.

lemma *sigmoid-symmetry*: *sigmoid* (*−x*) *= 1 − sigmoid x*
by (*smt* (*verit*, *ccfv-SIG*) *add-divide-distrib divide-self-if*
exp-ge-zero inverse-eq-divide sigmoid-alt-def sigmoid-def)

corollary *sigmoid*(*x*) *+ sigmoid*(*−x*) *= 1*
by (*simp add*: *sigmoid-symmetry*)

The sigmoid function is strictly increasing.

lemma *sigmoid-strictly-increasing*: *x1 < x2 ⟹ sigmoid x1 < sigmoid x2*
by (*unfold sigmoid-alt-def*,
smt (*verit*) *add-strict-left-mono divide-eq-0-iff exp-gt-zero exp-less-cancel-iff*
inverse-less-iff-less le-divide-eq-1-pos neg-0-le-iff-le neg-le-iff-le order-less-trans
real-add-le-0-iff)

lemma *sigmoid-at-zero*:
sigmoid 0 = 1/2
by (*simp add*: *sigmoid-def*)

lemma *sigmoid-left-dom-range*:
assumes *x < 0*
shows *sigmoid x < 1/2*
by (*metis assms sigmoid-at-zero sigmoid-strictly-increasing*)

lemma *sigmoid-right-dom-range*:
assumes *x > 0*
shows *sigmoid x > 1/2*
by (*metis assms sigmoid-at-zero sigmoid-strictly-increasing*)

2.2 Differentiability and Derivative Identities

Derivative: The derivative of the sigmoid function can be expressed in terms of itself:

$$\sigma'(x) = \sigma(x)\,(1 - \sigma(x)).$$

This identity is central to backpropagation for weight updates in neural networks, since it shows the derivative depends only on $\sigma(x)$, simplifying optimisation computations.

lemma *uminus-derive-minus-one*: (*uminus has-derivative* (∗) (−*1* :: *real*)) (*at a within A*)
  by (*rule has-derivative-eq-rhs*, (*rule derivative-intros*)+, *fastforce*)

lemma *sigmoid-differentiable*:
  (λ*x*. *sigmoid x*) *differentiable-on UNIV*
proof −
  have ∀ *x*. *sigmoid differentiable* (*at x*)
  proof
    fix *x* :: *real*
    have *num-diff*: (λ*x*. *exp x*) *differentiable* (*at x*)
    by (*simp add*: *field-differentiable-imp-differentiable field-differentiable-within-exp*)
    have *denom-diff*: (λ*x*. *1* + *exp x*) *differentiable* (*at x*)
      by (*simp add*: *num-diff*)
    hence (λ*x*. *exp x* / (*1* + *exp x*)) *differentiable* (*at x*)
    by (*metis add-le-same-cancel2 num-diff differentiable-divide exp-ge-zero not-one-le-zero*)

    thus *sigmoid differentiable* (*at x*)
      unfolding *sigmoid-def* **by** *simp*
  qed
  thus *?thesis*
    by (*simp add*: *differentiable-on-def*)
qed

lemma *sigmoid-differentiable*′:
 sigmoid field-differentiable at x
  by (*meson UNIV-I differentiable-on-def field-differentiable-def real-differentiableE sigmoid-differentiable*)

lemma *sigmoid-derivative*:
  shows *deriv sigmoid x* = *sigmoid x* ∗ (*1* − *sigmoid x*)
  unfolding *sigmoid-def*
proof −
  from *field-differentiable-within-exp*
  have *deriv* (λ*x*. *exp x* /(*1* + *exp x*)) *x* = (*deriv* (λ*x*. *exp x*) *x* ∗ (λ*x*. *1* + *exp x*) *x* − (λ*x*. *exp x*) *x* ∗ *deriv* (λ*x*. *1* + *exp x*) *x*) / ((λ*x*. *1* + *exp x*) *x*)^2
    by(*rule deriv-divide*,
      simp add: *Derivative.field-differentiable-add field-differentiable-within-exp*,
      smt (*verit*, *ccfv-threshold*) *exp-gt-zero*)
  also have ... = ((*exp x*) ∗ (*1* + *exp x*) −(*exp x*)∗ (*deriv* (λ*w*. ((λ*v*. *1*)*w* + (λ *u*.

exp u)*w*)) *x*)) / (*1* + *exp x*)^2
    by (*simp add*: *DERIV-imp-deriv*)
  also have ... = ((*exp x*) ∗ (*1* + *exp x*) −(*exp x*) ∗ (*deriv* (λ*v*. *1*) *x* + *deriv* (λ *u*. *exp u*) *x*)) / (*1* + *exp x*)^2
    by (*subst deriv-add*, *simp*, *simp add*: *field-differentiable-within-exp*, *auto*)
  also have ... = ((*exp x*) ∗ (*1* + *exp x*) −(*exp x*) ∗ (*exp x*)) / (*1* + *exp x*)^2
    by (*simp add*: *DERIV-imp-deriv*)
  also have ... = (*exp x* + (*exp x*)^2 −(*exp x*)^2) / (*1* + *exp x*)^2
    by (*simp add*: *ring-class.ring-distribs*(*1*))
  also have ... = (*exp x* / (*1* + *exp x*))∗(*1* / (*1* + *exp x*))
    by (*simp add*: *power2-eq-square*)
  also have ... = *exp x* / (*1* + *exp x*)∗(*1* − *exp x* / (*1* + *exp x*))
    by (*metis add.inverse-inverse inverse-eq-divide sigmoid-alt-def sigmoid-def sigmoid-symmetry*)
  finally show *deriv* (λ*x*. *exp x* / (*1* + *exp x*)) *x* = *exp x* / (*1* + *exp x*) ∗ (*1* − *exp x* / (*1* + *exp x*)).
qed

lemma *sigmoid-derivative*′: (*sigmoid has-real-derivative* (*sigmoid x* ∗ (*1* − *sigmoid x*))) (*at x*)
  by (*metis field-differentiable-derivI sigmoid-derivative sigmoid-differentiable*′)

lemma *deriv-one-minus-sigmoid*:
  deriv (λ*y*. *1* − *sigmoid y*) *x* = *sigmoid x* ∗ (*sigmoid x* − *1*)
  apply (*subst deriv-diff*)
    apply *simp*
  apply (*metis UNIV-I differentiable-on-def real-differentiableE sigmoid-differentiable field-differentiable-def*)
  apply (*metis deriv-const diff-0 minus-diff-eq mult-minus-right sigmoid-derivative*)
  done

2.3 Logit, Softmax, and the Tanh Connection

Logit (Inverse of Sigmoid): The inverse of the sigmoid function, often called the logit function, is defined by

$$\sigma^{-1}(y) \;=\; \ln\!\left(\tfrac{y}{1-y}\right), \quad 0 < y < 1.$$

This transformation converts a probability $y \in (0,1)$ (the output of the sigmoid) back into the corresponding log-odds.

definition *logit* :: *real* ⇒ *real* **where**
  logit p = (*if 0* < *p* ∧ *p* < *1 then ln* (*p* / (*1* − *p*)) *else undefined*)

lemma *sigmoid-logit-comp*:
  0 < *p* ∧ *p* < *1* ⟹ *sigmoid* (*logit p*) = *p*
proof −
  assume *0* < *p* ∧ *p* < *1*
  then show *sigmoid* (*logit p*) = *p*

by (*smt* (*verit*, *del-insts*) *divide-pos-pos exp-ln-iff logit-def real-shrink-Galois sigmoid-def*)
qed

lemma *logit-sigmoid-comp*:
logit (*sigmoid p*) = *p*
by (*smt* (*verit*, *best*) *sigmoid-less-1 sigmoid-logit-comp sigmoid-pos sigmoid-strictly-increasing*)

definition *softmax* :: *real^'k* ⇒ *real^'k* **where**
softmax z = (χ *i*. *exp* (*z* $ *i*) / (∑ *j*∈*UNIV*. *exp* (*z* $ *j*)))

lemma *tanh-sigmoid-relationship*:
2 ∗ *sigmoid* (*2* ∗ *x*) − *1* = *tanh x*
proof −
have *2* ∗ *sigmoid* (*2* ∗ *x*) − *1* = *2* ∗ (*1* / (*1* + *exp* (− (*2* ∗ *x*)))) − *1*
by (*simp add*: *inverse-eq-divide sigmoid-alt-def*)
also have ... = (*2* / (*1* + *exp* (− (*2* ∗ *x*)))) − *1*
by *simp*
also have ... = (*2* − (*1* + *exp* (− (*2* ∗ *x*)))) / (*1* + *exp* (− (*2* ∗ *x*)))
by (*smt* (*verit*, *ccfv-SIG*) *diff-divide-distrib div-self exp-gt-zero*)
also have ... = (*exp x* ∗ (*exp x* − *exp* (−*x*))) / (*exp x* ∗ (*exp x* + *exp* (−*x*)))
by (*smt* (*z3*) *exp-not-eq-zero mult-divide-mult-cancel-left-if tanh-altdef tanh-real-altdef*)
also have ... = (*exp x* − *exp* (−*x*)) / (*exp x* + *exp* (−*x*))
using *exp-gt-zero* **by** *simp*
also have ... = *tanh x*
by (*simp add*: *tanh-altdef*)
finally show *?thesis*.
qed

end

3 Derivative Identities and Smoothness

theory *Derivative-Identities-Smoothness*
imports *Sigmoid-Definition*
begin

Second derivative: The second derivative of the sigmoid function σ can be written as

$$\sigma''(x) = \sigma(x)\,(1-\sigma(x))\,(1-2\,\sigma(x)).$$

This identity is useful when analysing the curvature of σ, particularly in optimisation problems.

lemma *sigmoid-second-derivative*:
shows *Nth-derivative 2 sigmoid x* = *sigmoid x* ∗ (*1* − *sigmoid x*) ∗ (*1* − *2* ∗ *sigmoid x*)
proof −
have *Nth-derivative 2 sigmoid x* = *deriv* ((λ*w*. *deriv sigmoid w*)) *x*
by (*simp add*: *second-derivative-alt-def*)

also have ... = *deriv* ((λ*w*. (λ*a*. *sigmoid a*) *w* ∗ (((λ*u*.*1*) − (λ*v*. *sigmoid v*)) *w*))) *x*
  by (*simp add*: *sigmoid-derivative*)
also have ... = *sigmoid x* ∗ (*deriv* ((λ*u*.*1*) − (λ*v*. *sigmoid v*)) *x*) + *deriv* (λ*a*. *sigmoid a*) *x* ∗ ((λ*u*.*1*) − (λ*v*. *sigmoid v*)) *x*
  by (*rule deriv-mult*,
    simp add: *sigmoid-differentiable′*,
    simp add: *Derivative.field-differentiable-diff sigmoid-differentiable′*)
also have ... = *sigmoid x* ∗ (*deriv* (λ*y*. *1* − *sigmoid y*) *x*) + *deriv* (λ*a*. *sigmoid a*) *x* ∗ ((λ*u*.*1*) − (λ*v*. *sigmoid v*)) *x*
  by (*meson minus-apply*)
also have ... = *sigmoid x* ∗ (*deriv* (λ*y*. *1* − *sigmoid y*) *x*) + *deriv* (λ*a*. *sigmoid a*) *x* ∗ (λ*y*. *1* − *sigmoid y*) *x*
  by *simp*
also have ... = *sigmoid x* ∗ *sigmoid x* ∗ (*sigmoid x* −*1*) + *sigmoid x* ∗ (*1* − *sigmoid x*) ∗ (*1* − *sigmoid x*)
  by (*simp add*: *deriv-one-minus-sigmoid sigmoid-derivative*)
also have ... = *sigmoid x* ∗ (*1* − *sigmoid x*) ∗ (*1* − *2* ∗ *sigmoid x*)
  by (*simp add*: *right-diff-distrib*)
finally show *?thesis***.**
qed

Here we present the proof of the general nth derivative of the sigmoid function as given in the paper On the Derivatives of the Sigmoid by Ali A. Minai and Ronald D. Williams [4]. Their original derivation is natural and intuitive, guiding the reader step by step to the closed-form expression if one did not know it in advance. By contrast, our Isabelle formalisation assumes the final formula up front and then proves it directly by induction. Crucially, we make essential use of Stirling numbers of the second kindas formalised in the session Basic combinatorics in Isabelle/HOL (and the Archive of Formal Proofs) by Amine Chaieb, Florian Haftmann, Lukas Bulwahn, and Manuel Eberl.

theorem *nth-derivative-sigmoid*:
  ⋀*x*. *Nth-derivative n sigmoid x* =
    (∑ *k* = *1*..*n*+*1*. (−*1*) ^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*sigmoid x*) ^*k*)
proof (*induct n*)
case *0*
show *?case*
  by *simp*
next
fix *n x*
assume *induction-hypothesis*:
  ⋀*x*. *Nth-derivative n sigmoid x* =
    (∑ *k* = *1*..*n*+*1*. (−*1*) ^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*sigmoid x*) ^*k*)
show *Nth-derivative* (*Suc n*) *sigmoid x* =
    (∑ *k* = *1*..(*Suc n*)+*1*. (−*1*) ^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* ((*Suc n*)+*1*)

$k * (sigmoid\ x)\hat{}k)$
proof −

have *sigmoid-pwr-rule*: $\bigwedge k.\ deriv\ (\lambda v.\ (sigmoid\ v)\hat{}k)\ x = k * (sigmoid\ x)\hat{}(k - 1) * deriv\ (\lambda u.\ sigmoid\ u)\ x$
by (*subst deriv-pow*, *simp add*: *sigmoid-differentiable′*, *simp*)
have *index-shift*: $(\sum j = 1..n+1.\ ((-1)\hat{}(j+1+1) * fact\ (j - 1) * Stirling\ (n+1)\ j * j * ((sigmoid\ x)\hat{}(j+1)))) =$
$(\sum j = 2..n+2.\ (-1)\hat{}(j+1) * fact\ (j - 2) * Stirling\ (n+1)\ (j - 1) * (j - 1) * (sigmoid\ x)\hat{}j)$
by (*rule sum.reindex-bij-witness*[*of* - $\lambda j.\ j -1\ \lambda j.\ j + 1$], *simp-all*, *auto*)

have *simplified-terms*: $(\sum k = 1..n+1.\ ((-1)\hat{}(k+1) * fact\ (k - 1) * Stirling\ (n+1)\ k * k * (sigmoid\ x)\hat{}k) +$
$((-1)\hat{}(k+1) * fact\ (k - 2) * Stirling\ (n+1)\ (k-1) * (k-1) * (sigmoid\ x)\hat{}k)) =$
$(\sum k = 1..n+1.\ ((-1)\hat{}(k+1) * fact\ (k - 1) * Stirling\ (n+2)\ k * (sigmoid\ x)\hat{}k))$
proof −
have *equal-terms*: $\forall\ (k::nat) \geq 1.$
$((-1)\hat{}(k+1) * fact\ (k - 1) * Stirling\ (n+1)\ k * k * (sigmoid\ x)\hat{}k) +$
$((-1)\hat{}(k+1) * fact\ (k - 2) * Stirling\ (n+1)\ (k-1) * (k-1) * (sigmoid\ x)\hat{}k) =$
$((-1)\hat{}(k+1) * fact\ (k - 1) * Stirling\ (n+2)\ k * (sigmoid\ x)\hat{}k)$

proof(*clarify*)
fix *k*::*nat*
assume $1 \leq k$

have *real-of-int* $((-\ 1)\ \hat{}\ (k + 1) * fact\ (k - 1) * int\ (Stirling\ (n + 1)\ k) * int\ k) * sigmoid\ x\ \hat{}\ k +$
real-of-int $((-\ 1)\ \hat{}\ (k + 1) * fact\ (k - 2) * int\ (Stirling\ (n + 1)\ (k - 1)) * int\ (k - 1)) * sigmoid\ x\ \hat{}\ k =$
real-of-int $(((-\ 1)\ \hat{}\ (k + 1) * ((fact\ (k - 1) * int\ (Stirling\ (n + 1)\ k) * int\ k) +$
$(fact\ (k - 2) * int\ (Stirling\ (n + 1)\ (k - 1)) * int\ (k - 1))))) * sigmoid\ x\ \hat{}\ k$
by (*metis* (*mono-tags*, *opaque-lifting*) *ab-semigroup-mult-class.mult-ac*(*1*) *distrib-left mult.commute of-int-add*)
also have ... = *real-of-int* $(((-\ 1)\ \hat{}\ (k + 1) * ((fact\ (k - 1) * int\ (Stirling\ (n + 1)\ k) * int\ k) +$
$((int\ (k - 1) * fact\ (k - 2)) * int\ (Stirling\ (n + 1)\ (k - 1)))))) * sigmoid\ x\ \hat{}\ k$
by (*simp add*: *ring-class.ring-distribs*(*1*))
also have ... = *real-of-int* $(((-\ 1)\ \hat{}\ (k + 1) * ((fact\ (k - 1) * int\ (Stirling\ (n + 1)\ k) * int\ k) +$

(*fact* (*k* − *1*) ∗ *int* (*Stirling* (*n* + *1*) (*k* − *1*)))))) ∗ *sigmoid x* ^ *k*
by (*smt* (*verit*, *ccfv-threshold*) *Stirling.simps*(*3*) *add.commute diff-diff-left fact-num-eq-if mult-eq-0-iff of-nat-eq-0-iff one-add-one plus-1-eq-Suc*)
also have ... = *real-of-int* (((− *1*) ^ (*k* + *1*) ∗ *fact* (*k* − *1*)∗
(*Stirling* (*n* + *1*) *k* ∗ *k* + *Stirling* (*n* + *1*) (*k* − *1*)))) ∗ *sigmoid x* ^ *k*
by (*simp add*: *distrib-left*)
also have ... = *real-of-int* ((− *1*) ^ (*k* + *1*) ∗ *fact* (*k* − *1*) ∗ *int* (*Stirling* (*n* + *2*) *k*)) ∗ *sigmoid x* ^ *k*
by (*smt* (*z3*) *Stirling.simps*(*4*) *Suc-eq-plus1* ‹*1* ≤ *k*› *add.commute le-add-diff-inverse mult.commute nat-1-add-1 plus-nat.simps*(*2*))
finally show *real-of-int* ((− *1*) ^ (*k* + *1*) ∗ *fact* (*k* − *1*) ∗ *int* (*Stirling* (*n* + *1*) *k*) ∗ *int k*) ∗ *sigmoid x* ^ *k* +
real-of-int ((− *1*) ^ (*k* + *1*) ∗ *fact* (*k* − *2*) ∗ *int* (*Stirling* (*n* + *1*) (*k* − *1*)) ∗ *int* (*k* − *1*)) ∗ *sigmoid x* ^ *k* =
real-of-int ((− *1*) ^ (*k* + *1*) ∗ *fact* (*k* − *1*) ∗ *int* (*Stirling* (*n* + *2*) *k*)) ∗ *sigmoid x* ^ *k*.
qed
from *equal-terms* **show** *?thesis*
by *simp*
qed

have *Nth-derivative* (*Suc n*) *sigmoid x* = *deriv* (λ *w*. *Nth-derivative n sigmoid w*) *x*
by *simp*
also have ... = *deriv* (λ *w*.∑ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*sigmoid w*)^*k*) *x*
using *induction-hypothesis* **by** *presburger*
also have ... = (∑ *k* = *1*..*n*+*1*. *deriv* (λ*w*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*sigmoid w*)^*k*) *x*)
by (*rule deriv-sum*, *metis*(*mono-tags*) *DERIV-chain2 DERIV-cmult-Id field-differentiable-def field-differentiable-power sigmoid-differentiable*′)
also have ... = (∑ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *deriv* (λ*w*. (*sigmoid w*)^*k*) *x*)
by(*subst deriv-cmult*, *auto*, *simp add*: *field-differentiable-power sigmoid-differentiable*′)
also have ... = (∑ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^(*k* − *1*) ∗ *deriv* (λ*u*. *sigmoid u*) *x*))
using *sigmoid-pwr-rule* **by** *presburger*
also have ... = (∑ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^(*k* − *1*) ∗ (*sigmoid x* ∗ (*1* − *sigmoid x*))))
using *sigmoid-derivative* **by** *presburger*
also have ... = (∑ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ ((*sigmoid x*)^(*k* − *1*) ∗ (*sigmoid x*)^*1*) ∗ (*1* − *sigmoid x*)))
by (*simp add*: *mult.assoc*)
also have ... = (∑ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^(*k*−*1*+*1*) ∗ (*1* − *sigmoid x*)))

by (*metis* (*no-types*, *lifting*) *power-add*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k* ∗ (*1* −*sigmoid x*)))
by *fastforce*
also have ... = ($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) ∗ (*1* + −*sigmoid x*))
by (*simp add*: *ab-semigroup-mult-class.mult-ac*(*1*))
also have ... = ($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) ∗*1* +
(((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) ∗ (−*sigmoid x*)))
by (*meson vector-space-over-itself*.*scale-right-distrib*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) +
((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) ∗ (−*sigmoid x*))
by *simp*
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) +
($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) ∗ (−*sigmoid x*))
by (*metis* (*no-types*) *sum*.*distrib*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) +
($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *k* ∗ ((*sigmoid x*)^*k* ∗ (−*sigmoid x*))))
by (*simp add*: *mult*.*commute mult*.*left-commute*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) +
($\sum$ *j* = *1*..*n*+*1*. ((−*1*)^(*j*+*1*+*1*) ∗ *fact* (*j* − *1*) ∗ *Stirling* (*n*+*1*) *j* ∗ *j* ∗ ((*sigmoid x*)^(*j*+*1*))))
by (*simp add*: *mult*.*commute*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*k* ∗ (*sigmoid x*)^*k*)) +
($\sum$ *j* = *2*..*n*+*2*. (−*1*)^(*j*+*1*) ∗ *fact* (*j* − *2*) ∗ *Stirling* (*n*+*1*) (*j* − *1*) ∗ (*j* − *1*) ∗ (*sigmoid x*)^*j*)
using *index-shift* **by** *presburger*
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *k* ∗ (*sigmoid x*)^*k*) +
0 +
($\sum$ *j* = *2*..*n*+*2*. (−*1*)^(*j*+*1*) ∗ *fact* (*j* − *2*) ∗ *Stirling* (*n*+*1*) (*j* − *1*) ∗ (*j* − *1*) ∗ (*sigmoid x*)^*j*)
by (*smt* (*verit*, *ccfv-SIG*) *ab-semigroup-mult-class.mult-ac*(*1*) *of-int-mult of-int-of-nat-eq sum*.*cong*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *k* ∗ (*sigmoid x*)^*k*) +
((−*1*)^(*1*+*1*) ∗ *fact* (*1* − *2*) ∗ *Stirling* (*n*+*1*) (*1* − *1*) ∗ (*1* − *1*) ∗ (*sigmoid x*)^*1*) +
($\sum$ *k* = *2*..*n*+*2*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *2*) ∗ *Stirling* (*n*+*1*) (*k*

− *1*) ∗ (*k* −*1*) ∗ (*sigmoid x*)^*k*)
by *simp*
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *k* ∗ (*sigmoid x*)^*k*) +
($\sum$ *k* = *1*..*n*+*2*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *2*) ∗ *Stirling* (*n*+*1*) (*k*−*1*) ∗ (*k*−*1*) ∗ (*sigmoid x*)^*k*)
by (*smt* (*verit*) *Suc-eq-plus1 Suc-leI add-Suc-shift add-cancel-left-left cancel-comm-monoid-add-class.diff-cancel nat-1-add-1 of-nat-0 sum.atLeast-Suc-atMost zero-less-Suc*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *k* ∗ (*sigmoid x*)^*k*) +
($\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *2*) ∗ *Stirling* (*n*+*1*) (*k*−*1*) ∗ (*k*−*1*) ∗ (*sigmoid x*)^*k*) +
((−*1*)^((*n*+*2*)+*1*) ∗ *fact* ((*n*+*2*) − *2*) ∗ *Stirling* (*n*+*1*) ((*n*+*2*)−*1*) ∗ ((*n*+*2*)−*1*) ∗ (*sigmoid x*)^(*n*+*2*))
by *simp*
also have ... = ($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ *k* ∗ (*sigmoid x*)^*k*) +
((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *2*) ∗ *Stirling* (*n*+*1*) (*k*−*1*) ∗ (*k*−*1*) ∗ (*sigmoid x*)^*k*)) +
((−*1*)^((*n*+*2*)+*1*) ∗ *fact* ((*n*+*2*) − *2*) ∗ *Stirling* (*n*+*1*) ((*n*+*2*)−*1*) ∗ ((*n*+*2*)−*1*) ∗ (*sigmoid x*)^(*n*+*2*))
by (*metis* (*no-types*) *sum.distrib*)
also have ... = ($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*2*) *k* ∗ (*sigmoid x*)^*k*)) +
((−*1*)^((*n*+*2*)+*1*) ∗ *fact* ((*n*+*2*) − *2*) ∗ *Stirling* (*n*+*1*) ((*n*+*2*)−*1*) ∗ ((*n*+*2*)−*1*) ∗ (*sigmoid x*)^(*n*+*2*))
using *simplified-terms* **by** *presburger*
also have ... = ($\sum$ *k* = *1*..*n*+*1*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* ((*Suc n*) + *1*) *k* ∗ (*sigmoid x*)^*k*)) +
($\sum$ *k* = *Suc n* + *1*..*Suc n* + *1*.((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* ((*Suc n*) + *1*) *k* ∗ (*sigmoid x*)^(*k*)))
by(*subst atLeastAtMost-singleton*, *simp*)
also have ... = ($\sum$ *k* = *1*..(*Suc n*)+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* ((*Suc n*)+*1*) *k* ∗ (*sigmoid x*)^*k*)
by (*subst sum.cong*[**where** *B*={*1*..*n* + *1*}, **where** *h* = λ*k*. ((−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* ((*Suc n*) + *1*) *k* ∗ (*sigmoid x*)^(*k*))], *simp-all*)
finally show *?thesis***.**
qed
qed

corollary *nth-derivative-sigmoid-differentiable*:
Nth-derivative n sigmoid differentiable (*at x*)
proof −
have (λ*x*. $\sum$ *k* = *1*..*n*+*1*. (−*1*)^(*k*+*1*) ∗ *fact* (*k* − *1*) ∗ *Stirling* (*n*+*1*) *k* ∗ (*sigmoid x*)^*k*)
differentiable (*at x*)
proof −
have *differentiable-terms*: ⋀*k*. *1* ≤ *k* ∧ *k* ≤ *n*+*1* ⟹

(*λx. (−1)^(k+1) ∗ fact (k − 1) ∗ Stirling (n+1) k ∗ (sigmoid x)^k*) *differentiable (at x)*
proof(*clarify*)
fix *k ::nat*
assume *1 ≤ k*
assume *k ≤ n+1*
show (*λx. (−1)^(k+1) ∗ fact (k − 1) ∗ Stirling (n+1) k ∗ (sigmoid x)^k*) *differentiable (at x)*
by (*simp add: field-differentiable-imp-differentiable sigmoid-differentiable′*)
qed
then show *?thesis*
by(*subst differentiable-sum,simp+*)
qed
then show *?thesis*
using *nth-derivative-sigmoid* **by** *presburger*
qed

corollary *next-deriviative-sigmoid*: (*Nth-derivative n sigmoid has-real-derivative Nth-derivative (Suc n) sigmoid x*) (*at x*)
by (*simp add: DERIV-deriv-iff-real-differentiable nth-derivative-sigmoid-differentiable*)

corollary *deriv-sigmoid-has-deriv*: (*deriv sigmoid has-real-derivative deriv (deriv sigmoid) x*) (*at x*)
proof −
have ∀*f. Nth-derivative (Suc 0) f = deriv f*
using *Nth-derivative.simps(1,2)* **by** *presburger*
then show *?thesis*
by (*metis (no-types) DERIV-deriv-iff-real-differentiable nth-derivative-sigmoid-differentiable*)
qed

corollary *sigmoid-second-derivative′*:
(*deriv sigmoid has-real-derivative (sigmoid x ∗ (1 − sigmoid x) ∗ (1 − 2 ∗ sigmoid x)))* (*at x*)
using *deriv-sigmoid-has-deriv second-derivative-alt-def sigmoid-second-derivative*
by *force*

corollary *smooth-sigmoid*:
smooth-on sigmoid UNIV
unfolding *smooth-on-def*
by (*meson C-k-on-def differentiable-imp-continuous-on differentiable-on-def nth-derivative-sigmoid-differentia*
open-UNIV sigmoid-differentiable)

lemma *tendsto-exp-neg-at-infinity*: ((*λ(x :: real). exp (−x)*) ⟶ *0*) *at-top*
by *real-asymp*

end

4 Asymptotic and Qualitative Properties

```
theory Asymptotic-Qualitative-Properties
  imports Derivative-Identities-Smoothness
begin
```

4.1 Limits at Infinity of Sigmoid and its Derivative

— Asymptotic Behaviour — We have

$$\lim_{x\to+\infty} \sigma(x) = 1, \quad \lim_{x\to-\infty} \sigma(x) = 0.$$

```
lemma lim-sigmoid-infinity: ((λx. sigmoid x) ⟶ 1) at-top
  unfolding sigmoid-def by real-asymp

lemma lim-sigmoid-minus-infinity: (sigmoid ⟶ 0) at-bot
  unfolding sigmoid-def by real-asymp

lemma sig-deriv-lim-at-top: (deriv sigmoid ⟶ 0) at-top
proof (subst tendsto-at-top-epsilon-def, clarify)
  fix ε :: real
  assume ε-pos: 0 < ε
```

Using the fact that $\sigma(x) \to 1$ as $x \to +\infty$.

```
  obtain N where N-def: ∀x ≥ N. |sigmoid x − 1| < ε / 2
    using lim-sigmoid-infinity[unfolded tendsto-at-top-epsilon-def] ε-pos
    by (metis half-gt-zero)

  have deriv-bound: ∀x ≥ N. |deriv sigmoid x| ≤ |sigmoid x − 1|
  proof (clarify)
    fix x
    assume x ≥ N
    hence |deriv sigmoid x| = |sigmoid x − 1 + 1| ∗ |1 − sigmoid x|
      by (simp add: abs-mult sigmoid-derivative)

    also have ... ≤ |sigmoid x − 1|
      by (smt (verit) mult-cancel-right1 mult-right-mono sigmoid-range)
    finally show |deriv sigmoid x| ≤ |sigmoid x − 1|.
  qed

  have ∀x ≥ N. |deriv sigmoid x| < ε
  proof (clarify)
    fix x
    assume x ≥ N
    hence |deriv sigmoid x| ≤ |sigmoid x − 1|
      using deriv-bound by simp
    also have ... < ε / 2
      using ‹x ≥ N› N-def by simp
    also have ... < ε
```

using *ε-pos* **by** *simp*
finally show |*deriv sigmoid x*| < *ε* **.**
qed

then show ∃*N*::*real*. ∀*x*≥*N*. |*deriv sigmoid x* − (*0*::*real*)| < *ε*
by (*metis diff-zero*)
qed

lemma *sig-deriv-lim-at-bot*: (*deriv sigmoid* ⟶ *0*) *at-bot*
proof (*subst tendsto-at-bot-epsilon-def*, *clarify*)
fix *ε* :: *real*
assume *ε-pos*: *0* < *ε*

Using the fact that $\sigma(x) \to 0$ as $x \to -\infty$.

obtain *N* **where** *N-def*: ∀*x* ≤ *N*. |*sigmoid x* − *0*| < *ε* / *2*
using *lim-sigmoid-minus-infinity*[*unfolded tendsto-at-bot-epsilon-def*] *ε-pos*
by (*meson half-gt-zero*)

have *deriv-bound*: ∀*x* ≤ *N*. |*deriv sigmoid x*| ≤ |*sigmoid x* − *0*|
proof (*clarify*)
fix *x*
assume *x* ≤ *N*
hence |*deriv sigmoid x*| = |*sigmoid x* − *0* + *0*| ∗ |*1* − *sigmoid x*|
by (*simp add*: *abs-mult sigmoid-derivative*)
also have ... ≤ |*sigmoid x* − *0*|
by (*smt* (*verit*, *del-insts*) *mult-cancel-left2 mult-left-mono sigmoid-range*)
finally show |*deriv sigmoid x*| ≤ |*sigmoid x* − *0*|**.**
qed

have ∀*x* ≤ *N*. |*deriv sigmoid x*| < *ε*
proof (*clarify*)
fix *x*
assume *x* ≤ *N*
hence |*deriv sigmoid x*| ≤ |*sigmoid x* − *0*|
using *deriv-bound* **by** *simp*
also have ... < *ε* / *2*
using ‹*x* ≤ *N*› *N-def* **by** *simp*
also have ... < *ε*
using *ε-pos* **by** *simp*
finally show |*deriv sigmoid x*| < *ε***.**
qed

then show ∃*N*::*real*. ∀*x* ≤ *N*. |*deriv sigmoid x* − (*0*::*real*)| < *ε*
by (*metis diff-zero*)
qed

4.2 Curvature and Inflection

lemma *second-derivative-sigmoid-positive-on*:
assumes *x* < *0*

shows *Nth-derivative 2 sigmoid x > 0*
proof −
have *1 − 2 ∗ sigmoid x > 0*
using *assms sigmoid-left-dom-range* **by** *force*
then show *Nth-derivative 2 sigmoid x > 0*
by (*simp add*: *sigmoid-range sigmoid-second-derivative*)
qed

lemma *second-derivative-sigmoid-negative-on*:
assumes *x > 0*
shows *Nth-derivative 2 sigmoid x < 0*
proof −
have *1 − 2 ∗ sigmoid x < 0*
by (*smt* (*verit*) *assms sigmoid-strictly-increasing sigmoid-symmetry*)
then show *Nth-derivative 2 sigmoid x < 0*
by (*simp add*: *mult-pos-neg sigmoid-range sigmoid-second-derivative*)
qed

lemma *sigmoid-inflection-point*:
Nth-derivative 2 sigmoid 0 = 0
by (*simp add*: *sigmoid-alt-def sigmoid-second-derivative*)

4.3 Monotonicity and Bounds of the First Derivative

lemma *sigmoid-positive-derivative*:
deriv sigmoid x > 0
by (*simp add*: *sigmoid-derivative sigmoid-range*)

lemma *sigmoid-deriv-0*:
deriv sigmoid 0 = 1/4
proof −
have *f1*: *1 / (1 + 1) = sigmoid 0*
by (*simp add*: *sigmoid-def*)
then have *f2*: *∀ r. sigmoid 0 ∗ (r + r) = r*
by *simp*
then have *f3*: *∀ n. sigmoid 0 ∗ numeral (num.Bit0 n) = numeral n*
by (*metis* (*no-types*) *numeral-Bit0*)
have *f4*: *∀ r. sigmoid r ∗ sigmoid (− r) = deriv sigmoid r*
using *sigmoid-derivative sigmoid-symmetry* **by** *presburger*
have *sigmoid 0 = 0 ⟶ deriv sigmoid 0 = 1 / 4*
using *f1* **by** *force*
then show *?thesis*
using *f4 f3 f2* **by** (*metis* (*no-types*) *add.inverse-neutral divide-divide-eq-right nonzero-mult-div-cancel-left one-add-one zero-neq-numeral*)
qed

lemma *deriv-sigmoid-increase-on-negatives*:
assumes *x2 < 0*
assumes *x1 < x2*

shows *deriv sigmoid x1* < *deriv sigmoid x2*
by(*rule DERIV-pos-imp-increasing*, *simp add*: *assms*(*2*), *metis assms*(*1*) *deriv-sigmoid-has-deriv*
dual-order.strict-trans linorder-not-le nle-le second-derivative-alt-def second-derivative-sigmoid-positive-on)

lemma *deriv-sigmoid-decreases-on-positives*:
assumes *0* < *x1*
assumes *x1* < *x2*
shows *deriv sigmoid x2* < *deriv sigmoid x1*
by(*rule DERIV-neg-imp-decreasing*, *simp add*: *assms*(*2*), *metis assms*(*1*) *deriv-sigmoid-has-deriv*
dual-order.strict-trans linorder-not-le nle-le second-derivative-alt-def second-derivative-sigmoid-negative-on)

lemma *sigmoid-derivative-upper-bound*:
assumes *x*≠ *0*
shows *deriv sigmoid x* < *1/4*
proof(*cases x* ≤ *0*)
assume *x*≤*0*
then have *neg-case*: *x* < *0*
using *assms* **by** *linarith*
then have *deriv sigmoid x* < *deriv sigmoid 0*
proof(*rule DERIV-pos-imp-increasing-open*)
show ⋀*xa*::*real*. *x* < *xa* ⟹ *xa* < *0* ⟹ ∃ *y*::*real*. (*deriv sigmoid has-real-derivative y*) (*at xa*) ∧ *0* < *y*
by (*metis* (*no-types*) *deriv-sigmoid-has-deriv second-derivative-alt-def second-derivative-sigmoid-positive-on*)
show *continuous-on* {*x*..*0*::*real*} (*deriv sigmoid*)
by (*meson DERIV-atLeastAtMost-imp-continuous-on deriv-sigmoid-has-deriv*)
qed
then show *deriv sigmoid x* < *1/4*
by (*simp add*: *sigmoid-deriv-0*)
next
assume ¬ *x* ≤ *0*
then have *0* < *x*
by *linarith*
then have *deriv sigmoid x* < *deriv sigmoid 0*
proof(*rule DERIV-neg-imp-decreasing-open*)
show ⋀*xa*::*real*. *0* < *xa* ⟹ *xa* < *x* ⟹ ∃ *y*::*real*. (*deriv sigmoid has-real-derivative y*) (*at xa*) ∧ *y* < *0*
by (*metis* (*no-types*) *deriv-sigmoid-has-deriv second-derivative-alt-def second-derivative-sigmoid-negative-on*)
show *continuous-on* {*0*..*x*::*real*} (*deriv sigmoid*)
by (*meson DERIV-atLeastAtMost-imp-continuous-on deriv-sigmoid-has-deriv*)
qed
then show *deriv sigmoid x* < *1/4*
by (*simp add*: *sigmoid-deriv-0*)
qed

corollary *sigmoid-derivative-range*:
0 < deriv sigmoid x ∧ deriv sigmoid x ≤ 1/4
by (*smt* (*verit*, *best*) *sigmoid-deriv-0 sigmoid-derivative-upper-bound sigmoid-positive-derivative*)

4.4 Sigmoidal and Heaviside Step Functions

definition *sigmoidal* :: (*real* ⇒ *real*) ⇒ *bool* **where**
sigmoidal f ≡ (*f* ⟶ *1*) *at-top* ∧ (*f* ⟶ *0*) *at-bot*

lemma *sigmoid-is-sigmoidal*: *sigmoidal sigmoid*
unfolding *sigmoidal-def*
by (*simp add*: *lim-sigmoid-infinity lim-sigmoid-minus-infinity*)

definition *heaviside* :: *real* ⇒ *real* **where**
heaviside x = (*if x < 0 then 0 else 1*)

lemma *heaviside-right*: *x ≥ 0* ⟹ *heaviside x = 1*
by (*simp add*: *heaviside-def*)

lemma *heaviside-left*: *x < 0* ⟹ *heaviside x = 0*
by (*simp add*: *heaviside-def*)

lemma *heaviside-mono*: *x < y* ⟹ *heaviside x ≤ heaviside y*
by (*simp add*: *heaviside-def*)

lemma *heaviside-limit-neg-infinity*:
(*heaviside* ⟶ *0*) *at-bot*
by(*rule tendsto-eventually*, *subst eventually-at-bot-dense*, *meson heaviside-def*)

lemma *heaviside-limit-pos-infinity*:
(*heaviside* ⟶ *1*) *at-top*
by(*rule tendsto-eventually*, *subst eventually-at-top-dense*, *meson heaviside-def order.asym*)

lemma *heaviside-is-sigmoidal*: *sigmoidal heaviside*
by (*simp add*: *heaviside-limit-neg-infinity heaviside-limit-pos-infinity sigmoidal-def*)

4.5 Uniform Approximation by Sigmoids

lemma *sigmoidal-uniform-approximation*:
assumes *sigmoidal σ*
assumes (*ε* :: *real*) > *0* **and** (*h* :: *real*) > *0*
shows ∃ (*ω*::*real*)>*0*. ∀ *w*≥*ω*. ∀ *k*<*length* (*xs* :: *real list*).
(∀ *x*. *x − xs!k ≥ h* ⟶ |*σ* (*w* ∗ (*x − xs!k*)) − *1*| < *ε*) ∧
(∀ *x*. *x − xs!k ≤ −h* ⟶ |*σ* (*w* ∗ (*x − xs!k*))| < *ε*)
proof −

By the sigmoidal assumption, we extract the limits

$$\lim_{x\to+\infty} \sigma(x) = 1 \quad \text{(limit at\_top)} \quad \text{and} \quad \lim_{x\to-\infty} \sigma(x) = 0 \quad \text{(limit at\_bot)}.$$

have *lim-at-top*: (σ ⟶ *1*) *at-top*
 using *assms*(*1*) **unfolding** *sigmoidal-def* **by** *simp*
then obtain *Ntop* **where** *Ntop-def*: $\forall x \geq Ntop.\ |\sigma\ x - 1| < \varepsilon$
 using *assms*(*2*) *tendsto-at-top-epsilon-def* **by** *blast*

have *lim-at-bot*: (σ ⟶ *0*) *at-bot*
 using *assms*(*1*) **unfolding** *sigmoidal-def* **by** *simp*
then obtain *Nbot* **where** *Nbot-def*: $\forall x \leq Nbot.\ |\sigma\ x| < \varepsilon$
 using *assms*(*2*) *tendsto-at-bot-epsilon-def* **by** *fastforce*

Define ω to control the approximation.

obtain ω **where** ω*-def*: $\omega = max\ (max\ 1\ (Ntop\ /\ h))\ (-Nbot\ /\ h)$
 by *blast*
then have ω*-pos*: $0 < \omega$ **using** *assms*(*2*) **by** *simp*

Show that ω satisfies the required property.

show *?thesis*
proof (*intro exI*[**where** $x = \omega$] *allI impI conjI insert* ω*-pos*)
 fix *w* :: *real* **and** *k* :: *nat* **and** *x* :: *real*
 assume *w-ge-*ω: $\omega \leq w$
 assume *k-bound*: $k < length\ xs$

Case 1: $x - xs!k \geq h$.

 have $w * h \geq Ntop$
  using ω*-def assms*(*3*) *pos-divide-le-eq w-ge-*ω **by** *auto*

 then show $x - xs!k \geq h \Longrightarrow |\sigma\ (w * (x - xs!k)) - 1| < \varepsilon$
  using *Ntop-def*
  by (*smt* (*verit*) ω*-pos mult-less-cancel-left w-ge-*ω)

Case 2: $x - xs!k \leq -h$.

 have $-w * h \leq Nbot$
  using ω*-def assms*(*3*) *pos-divide-le-eq w-ge-*ω
  by (*smt* (*verit*, *ccfv-SIG*) *mult-minus-left*)
 then show $x - xs!k \leq -h \Longrightarrow |\sigma\ (w * (x - xs!k))| < \varepsilon$
  using *Nbot-def*
  by (*smt* (*verit*, *best*) ω*-pos minus-mult-minus mult-less-cancel-left w-ge-*ω)
qed
qed

end

5 Universal Approximation Theorem

theory *Universal-Approximation*

imports *Asymptotic-Qualitative-Properties*
begin

In this theory, we formalize the Universal Approximation Theorem (UAT) for continuous functions on a closed interval $[a, b]$. The theorem states that any continuous function $f\colon [a, b] \to \mathbb{R}$ can be uniformly approximated by a finite linear combination of shifted and scaled sigmoidal functions. The classical result was first proved by Cybenko [3] and later constructively by Costarelli and Spigler [2], the latter approach forms the basis of our formalization. Their paper is available online at https://link.springer.com/article/10.1007/s10231-013-0378-y.

lemma *uniform-continuity-interval*:
  fixes f :: *real* ⇒ *real*
  assumes $a < b$
  assumes *continuous-on* $\{a..b\}$ f
  assumes $\varepsilon > 0$
shows $\exists\,\delta{>}0.\ (\forall\, x\ y.\ x \in \{a..b\} \wedge y \in \{a..b\} \wedge |x - y| < \delta \longrightarrow |f\,x - f\,y| < \varepsilon)$
proof −
  have *uniformly-continuous-on* $\{a..b\}$ f
    using *assms(1,2) compact-uniformly-continuous* **by** *blast*
  thus *?thesis*
    unfolding *uniformly-continuous-on-def*
    by (*metis assms(3) dist-real-def*)
qed

definition *bounded-function* :: (*real* ⇒ *real*) ⇒ *bool* **where**
  bounded-function $f \longleftrightarrow$ *bdd-above* (*range* ($\lambda x.\ |f\,x|$))

definition *unif-part* :: *real* ⇒ *real* ⇒ *nat* ⇒ *real list* **where**
  unif-part $a\ b\ N$ =
    map ($\lambda k.\ a + (real\ k - 1\ ) * ((b - a)\ /\ real\ N\ ))\ [0..{<}N{+}2]$

value *unif-part* $(0{::}real)\ 1\ 4$

theorem *sigmoidal-approximation-theorem*:
  assumes *sigmoidal-function*: *sigmoidal* σ
  assumes *bounded-sigmoidal*: *bounded-function* σ
  assumes *a-lt-b*: $a < b$
  assumes *contin-f*: *continuous-on* $\{a..b\}$ f
  assumes *eps-pos*: $0 < \varepsilon$
  defines $xs\ N \equiv$ *unif-part* $a\ b\ N$
  shows $\exists\, N{::}nat.\ \exists\,(w{::}real) > 0.(N > 0) \wedge$
      $(\forall\, x \in \{a..b\}.$
        $|(\sum k{\in}\{2..N{+}1\}.\ (f(xs\ N\ !\ k) - f(xs\ N\ !\ (k - 1))) * \sigma(w * (x - xs$
$N\ !\ k)))$
          $+ f(a) * \sigma(w * (x - xs\ N\ !\ 0)) - f\,x| < \varepsilon)$
proof−

obtain η **where** *η-def*: $\eta = \varepsilon$ / ((*Sup* ((λx. $|f\ x|$) ' {*a..b*})) + (*2* ∗ (*Sup* ((λx. $|\sigma\ x|$) ' *UNIV*))) + *2*)
by *blast*

have *η-pos*: $\eta > 0$
unfolding *η-def*
proof −
have *sup-abs-nonneg*: *Sup* ((λx. $|f\ x|$) ' {*a..b*}) ≥ *0*
proof −
have ∀ *x* ∈ {*a..b*}. $|f\ x|$ ≥ *0*
by *simp*
hence *bdd-above* ((λx. $|f\ x|$) ' {*a..b*})
by (*metis a-lt-b bdd-above-Icc contin-f continuous-image-closed-interval continuous-on-rabs order-less-le*)
thus *?thesis*
by (*meson a-lt-b abs-ge-zero atLeastAtMost-iff cSUP-upper2 order-le-less*)
qed

have *sup-σ-nonneg*: *Sup* ((λx. $|\sigma\ x|$) ' *UNIV*) ≥ *0*
proof −
have ∀ *x* ∈ {*a..b*}. $|\sigma\ x|$ ≥ *0*
by *simp*
hence *bdd-above* ((λx. $|\sigma\ x|$) ' *UNIV*)
using *bounded-function-def bounded-sigmoidal* **by** *presburger*
thus *?thesis*
by (*meson abs-ge-zero cSUP-upper2 iso-tuple-UNIV-I*)
qed

obtain *denom* **where** *denom-def*: *denom* = (*Sup* ((λx. $|f\ x|$) ' {*a..b*})) + (*2* ∗ (*Sup* ((λx. $|\sigma\ x|$) ' *UNIV*))) + *2*
by *blast*
have *denom-pos*: *denom* > *0*
proof −
have *two-sup-σ-nonneg*: *0* ≤ *2* ∗ (*Sup* ((λx. $|\sigma\ x|$) ' *UNIV*))
by(*rule mult-nonneg-nonneg*, *simp*, *simp add*: *sup-σ-nonneg*)
have *0* ≤ (*Sup* ((λx. $|f\ x|$) ' {*a..b*})) + *2* ∗ (*Sup* ((λx. $|\sigma\ x|$) ' *UNIV*))
by(*rule add-nonneg-nonneg*, *smt sup-abs-nonneg*, *smt two-sup-σ-nonneg*)
then have *denom* ≥ *2* **unfolding** *denom-def*
by *linarith*
thus *denom* > *0* **by** *linarith*
qed
then show *0* < ε / ((*SUP x* ∈ {*a..b*}. $|f\ x|$) + *2* ∗ (*SUP x* ∈ *UNIV* . $|\sigma\ x|$) + *2*)
using *eps-pos sup-σ-nonneg sup-abs-nonneg* **by** *auto*
qed

have ∃ δ>*0*. ∀ *x y*. *x* ∈ {*a..b*} ∧ *y* ∈ {*a..b*} ∧ $|x - y| < \delta$ ⟶ $|f\ x - f\ y| < \eta$
by(*rule uniform-continuity-interval*,(*simp add*: *assms*(*3*,*4*))+, *simp add*: *η-pos*)

then obtain δ **where** *δ-pos*: $\delta > 0$
and *δ-prop*: $\forall x \in \{a..b\}.\ \forall y \in \{a..b\}.\ |x - y| < \delta \longrightarrow |f\ x - f\ y| < \eta$
by *blast*

obtain N **where** *N-def*: $N = (nat\ (\lfloor max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta))\rfloor) + 1)$
by *simp*

have *N-defining-properties*: $N > 2 * (b - a) / \delta \wedge N > 3 \wedge N > 1 / \eta$
unfolding *N-def*
proof −
have $max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta)) \geq 2 * (b - a) / \delta \wedge$
$max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta)) \geq 2 \wedge$
$max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta)) \geq 1 / \eta$
unfolding *max-def* **by** *simp*
then show $2 * (b - a) / \delta < nat\ \lfloor max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta))\rfloor + 1 \wedge$
$3 < nat\ \lfloor max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta))\rfloor + 1 \wedge$
$1 / \eta < nat\ \lfloor max\ 3\ (max\ (2 * (b - a) / \delta)\ (1 / \eta))\rfloor + 1$
by (*smt* (*verit*, *best*) *floor-le-one numeral-Bit1 numeral-less-real-of-nat-iff numeral-plus-numeral of-nat-1 of-nat-add of-nat-nat one-plus-numeral real-of-int-floor-add-one-gt*)
qed
then have *N-gt-3*: $N > 3$
by *simp*
then have *N-pos*: $N > 0$
by *simp*

obtain h **where** *h-def*: $h = (b-a)/N$
by *simp*
then have *h-pos*: $h > 0$
using *N-defining-properties a-lt-b* **by** *force*

have *h-lt-δ-half*: $h < \delta / 2$
proof −
have $N > 2 * (b - a) / \delta$
using *N-defining-properties* **by** *force*
then have $N / 2 > (b - a) / \delta$
by (*simp add*: *mult.commute*)
then have $(N / 2) * \delta > (b - a)$
by (*smt* (*verit*, *ccfv-SIG*) *δ-pos divide-less-cancel nonzero-mult-div-cancel-right*)
then have $(\delta / 2) * N > (b - a)$
by (*simp add*: *mult.commute*)
then have $(\delta / 2) > (b - a) / N$
by (*smt* (*verit*, *ccfv-SIG*) *δ-pos a-lt-b divide-less-cancel nonzero-mult-div-cancel-right zero-less-divide-iff*)

then show *h* < δ / *2*
using *h-def* **by** *blast*
qed

have *one-over-N-lt-eta*: *1* / *N* < η
proof −
have *f1*: *real N* ≥ *max* (*2* ∗ (*b* − *a*) / δ − *1*) (*1* / η)
unfolding *N-def* **by** *linarith*
have *real N* ≥ *1* / η
unfolding *max-def* **using** *f1 max.bounded-iff* **by** *blast*
hence *f2*: *1* / *real N* ≤ η
using *η-pos* **by** (*smt* (*verit*, *ccfv-SIG*) *divide-divide-eq-right le-divide-eq-1 mult.commute zero-less-divide-1-iff*)
then show *1* / *real N* < η
using *N-defining-properties nle-le* **by** *fastforce*
qed

have *xs-eqs*: *xs N* = *map* (λ*k*. *a* + (*real k* − *1*) ∗ ((*b* − *a*) / *N*)) [*0*..<*N*+*2*]
using *unif-part-def xs-def* **by** *presburger*

then have *xs-els*: ⋀*k*. *k* ∈ {*0*..*N*+*1*} ⟶ *xs N* ! *k* = *a* + (*real k*−*1*) ∗ *h*
by (*metis* (*no-types*, *lifting*) *Suc-1 add-0 add-Suc-right atLeastAtMost-iff diff-zero h-def linorder-not-le not-less-eq-eq nth-map-upt*)

have *zeroth-element*: *xs N* !*0* = *a*−*h*
by (*simp add*: *xs-els*)
have *first-element*: *xs N* !*1* = *a*
by (*simp add*: *xs-els*)
have *last-element*: *xs N* !(*N*+*1*) = *b*
proof −
have *xs N* !(*N*+*1*) = *a* + *N* ∗ *h*
using *xs-els* **by** *force*
then show *?thesis*
by (*simp add*: *N-pos h-def*)
qed

have *difference-of-terms*: ⋀*j k* . *j* ∈ {*1*..*N*+*1*} ∧ *k* ∈ {*1*..*N*+*1*} ∧ *j*≤ *k* ⟶ *xs N* ! *k* − *xs N* ! *j* = *h*∗(*real k*−*j*)
proof(*clarify*)
fix *j k*
assume *j-type*: *j* ∈ {*1*..*N* + *1*}
assume *k-type*: *k* ∈ {*1*..*N* + *1*}
assume *j-leq-k*: *j* ≤ *k*

have *j-th-el*: $xs\ N\ !\ j\ = (a + \ (real\ j-1) * h)$
  using *j-type xs-els* **by** *auto*
have *k-th-el*: $xs\ N\ \ !\ k\ = (a + \ (real\ k-1) * h)$
  using *k-type xs-els* **by** *auto*
then show $xs\ N\ !\ k - xs\ N\ !\ j = h * (real\ k - j)$
  by (*smt* (*verit*, *del-insts*) *j-th-el left-diff-distrib′ mult.commute*)
qed
then have *difference-of-adj-terms*: $\bigwedge k\ .\ \ k \in \{1..N+1\} \ \longrightarrow xs\ N\ !\ k - xs\ N\ !\ (k-1) = h$
proof −
fix $k :: nat$
have $k = 1 \longrightarrow k \in \{1..N + 1\} \longrightarrow xs\ N\ !\ k - xs\ N\ !\ (k - 1) = h$
  using *first-element zeroth-element* **by** *auto*
then show $k \in \{1..N + 1\} \longrightarrow xs\ N\ !\ k - xs\ N\ !\ (k - 1) = h$
  using *difference-of-terms le-diff-conv* **by** *fastforce*
qed
have *adj-terms-lt*: $\bigwedge k\ .\ \ k \in \{1..N+1\} \ \longrightarrow \ |xs\ N\ !\ k - xs\ N\ !\ (k - 1)| < \delta$
proof(*clarify*)
fix k
assume *k-type*: $k \in \{1..N + 1\}$
then have $|xs\ N\ !\ k - xs\ N\ !\ (k - 1)| = h$
  using *difference-of-adj-terms h-pos* **by** *auto*
also have ... $< \delta\ /2$
  using *h-lt-δ-half* **by** *auto*
also have ... $< \delta$
  by (*simp add*: *δ-pos*)
finally show $|xs\ N\ !\ k - xs\ N\ \ !\ (k - 1)| < \delta$**.**
qed

from *difference-of-terms* **have** *list-increasing*: $\bigwedge j\ \ k\ \ .\ j \in \{1..N+1\} \wedge k \in \{1..N+1\} \wedge j \le k \longrightarrow \ \ xs\ N\ !\ j \le xs\ N\ !k$
  by (*smt* (*verit*, *ccfv-SIG*) *h-pos of-nat-eq-iff of-nat-mono zero-less-mult-iff*)
have *els-in-ab*: $\bigwedge k.\ k \in \{1..N+1\} \longrightarrow xs\ N\ !\ k \in \{a..b\}$
  using *first-element last-element list-increasing* **by** *force*

from *sigmoidal-function N-pos h-pos* **have** $\exists\,\omega > 0.\ \forall\,w \ge \omega.\ \forall\,k < length\ (xs\ N).$
$(\forall\,x.\ x - xs\ N\ !k \ge h \ \ \longrightarrow |\sigma\ (w * (x - xs\ N\ !k)) - 1| < 1/N) \wedge$
$(\forall\,x.\ x - xs\ N!k \le -h \ \longrightarrow |\sigma\ (w * (x - xs\ N!k))| < 1/N)$
  by(*subst sigmoidal-uniform-approximation*, *simp-all*)
then obtain ω **where** *ω-pos*: $\omega > 0$
  and *ω-prop*: $\forall\,w \ge \omega.\ \forall\,k < length\ (xs\ N).$
$(\forall\,x.\ x - xs\ N\ !k \ge h \longrightarrow |\sigma\ (w * (x - xs\ N\ !k)) - 1| < 1/N) \wedge$
$(\forall\,x.\ x - xs\ N\ !k \le -h \longrightarrow |\sigma\ (w * (x - xs\ N!k))| < 1/N)$

by *blast*
then obtain *w* **where** *w-def*: $w \geq \omega$ **and** *w-prop*: $\forall\, k < length\ (xs\ N)$.
$(\forall\, x.\ x - xs\ N\ !k \geq h \longrightarrow |\sigma\ (w * (x - xs\ N\ !k)) - 1| < 1/N) \wedge$
$(\forall\, x.\ x - xs\ N\ !k \leq -h \longrightarrow |\sigma\ (w * (x - xs\ N!k))| < 1/N)$
and *w-pos*: $w > 0$
by *auto*

obtain *G-Nf* **where** *G-Nf-def*:
$G\text{-}Nf \equiv (\lambda x.$
$(\sum k\in\{2..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$+ f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0)))$
by *blast*

show $\exists\, N\, w.\ 0 < w \wedge 0 < N \wedge (\forall\, x\in\{a..b\}.\ |(\sum k = 2..N + 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) + f\ a * \sigma\ (w * (x - xs\ N\ !\ 0)) - f\ x| < \varepsilon)$
proof (*intro exI*[**where** *x=N*] *exI*[**where** *x=w*] *conjI allI impI insert w-pos N-pos xs-def*, *safe*)
fix *x::real*
assume *x-in-ab*: $x \in \{a..b\}$

have $\exists\, i.\ i \in \{1..N\} \wedge x \in \{xs\ N\ !\ i\ ..\ xs\ N\ !\ (i+1)\}$
proof −
have *intervals-cover*: $\{xs\ N\ !\ 1\ ..\ xs\ N\ !\ (N+1)\} \subseteq (\bigcup i\in\{1..N\}.\ \{xs\ N!\ i\ ..\ xs\ N!\ (i+1)\})$
proof
fix *x::real*
assume *x-def*: $x \in \{xs\ N!\ 1\ ..\ xs\ N\ !\ (N+1)\}$
then have *lower-bound*: $x \geq xs\ N\ !\ 1$
by *simp*
from *x-def* **have** *upper-bound*: $x \leq xs\ N!\ (N+1)$
by *simp*

obtain *j* **where** *j-def*: $j = (GREATEST\ j.\ xs\ N\ !\ j \leq x \wedge\ j \in \{1..N+1\})$

```
        by blast
      have nonempty-definition: {j ∈ {1..N+1}. xs N ! j ≤ x} ≠ {}
        using lower-bound by force
      then have j-exists:∃ j ∈ {1..N+1}. xs N ! j ≤ x
        by blast
      then have j-bounds: j ∈ {1..N+1}
        by (smt (verit) GreatestI-nat  atLeastAtMost-iff j-def)
      have xs-j-leq-x: xs N ! j ≤ x
     by (metis (mono-tags, lifting) GreatestI-ex-nat atLeastAtMost-iff empty-Collect-eq
j-def
           nonempty-definition)

      show x ∈ (⋃ i ∈ {1..N}. {xs N ! i..xs N ! (i + 1)})
      proof(cases j = N+1)
        show j = N + 1 ⟹ x ∈ (⋃ i ∈ {1..N}. {xs N ! i..xs N ! (i + 1)})
          using N-pos els-in-ab last-element upper-bound xs-j-leq-x by force
      next
        assume j-not-SucN:j ≠ N + 1
        then have j-type: j ∈ {1..N}
          by (metis Suc-eq-plus1 atLeastAtMost-iff j-bounds le-Suc-eq)
        then have Suc-j-type: j + 1 ∈ {2..N+1}
          by (metis Suc-1 Suc-eq-plus1 atLeastAtMost-iff diff-Suc-Suc diff-is-0-eq)
         have equal-sets: {j ∈ {1..N+1}. xs N ! j ≤ x} = {j ∈ {1..N}. xs N ! j
≤ x}
        proof
          show {j ∈ {1..N}. xs N ! j ≤ x} ⊆ {j ∈ {1..N + 1}. xs N ! j ≤ x}
            by auto
          show {j ∈ {1..N + 1}. xs N ! j ≤ x} ⊆ {j ∈ {1..N}. xs N ! j ≤ x}
        by (safe, metis (no-types, lifting) Greatest-equality Suc-eq-plus1 j-not-SucN
atLeastAtMost-iff j-def le-Suc-eq)
        qed

        have xs-j1-not-le-x: ¬ (xs N ! (j+1) ≤ x)
        proof(rule ccontr)
          assume BWOC: ¬ ¬ xs N ! (j + 1) ≤ x
          then have Suc-j-type':j+1 ∈ {1..N}
            using Suc-j-type equal-sets add.commute by auto
          from j-def show False
              using equal-sets
                   by (smt (verit, del-insts) BWOC Greatest-le-nat One-nat-def
Suc-eq-plus1 Suc-j-type' Suc-n-not-le-n  atLeastAtMost-iff mem-Collect-eq)
        qed
        then have x ∈ {xs N ! j .. xs N ! (j+1)}
          by (simp add: xs-j-leq-x)
        then show ?thesis
          using j-type by blast
      qed
    qed
    then show ?thesis
```

using *first-element last-element x-in-ab* **by** *fastforce*
qed
then obtain *i* **where** *i-def*: $i \in \{1..N\} \wedge x \in \{xs\ N\ !\ i\ ..\ xs\ N\ !\ (i+1)\}$
by *blast*
then have *i-ge-1*: $i \geq 1$
using *atLeastAtMost-iff* **by** *blast*

have *i-leq-N*: $i \leq N$
using *i-def* **by** *presburger*
then have *xs-i*: $xs\ N\ !\ i = a + (real\ i - 1) * h$
using *xs-els* **by** *force*
have *xs-Suc-i*: $xs\ N\ !\ (i + 1) = a + real\ i * h$
proof −
have $(i+1) \in \{0..N+1\} \longrightarrow xs\ N\ !\ (i+1) = a + (real\ (i+1) - 1) * h$
using *xs-els* **by** *blast*
then show *?thesis*
using *i-leq-N* **by** *fastforce*
qed

from *i-def* **have** *x-lower-bound-aux*: $x \geq\ (xs\ N\ !\ i)$
using *atLeastAtMost-iff* **by** *blast*
then have *x-lower-bound*: $x \geq a + real\ (i-1) * h$
by (*metis xs-i i-ge-1 of-nat-1 of-nat-diff*)

from *i-def* **have** *x-upper-bound-aux*: $xs\ N!\ (i+1) \geq\ x$
using *atLeastAtMost-iff* **by** *blast*
then have *x-upper-bound*: $a + real\ i * h \geq x$
using *xs-Suc-i* **by** *fastforce*

obtain *L* **where** *L-def*:
$\bigwedge i.\ L\ i = (if\ i = 1 \vee i = 2\ then$
$(\lambda x.\ f(a) + (f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3)) +$
$(f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2)))$
else
$(\lambda x.\ (\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))) + f(a) +$
$(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i-1))) * \sigma\ (w * (x - xs\ N\ !\ i)) +$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ i)) * \sigma\ (w * (x - xs\ N\ !\ (i+1)))))$
by *force*

obtain *I-1* **where** *I-1-def*: $\bigwedge i.1 \leq i \wedge i \leq N \longrightarrow I\text{-}1\ i =\ (\lambda x.\ |G\text{-}Nf\ x - L\ i$
$x|)$
by *force*

obtain *I-2* **where** *I-2-def*: $\bigwedge i.\ 1 \leq i \wedge i \leq N \longrightarrow I\text{-}2\ i = (\lambda x.\ |L\ i\ x - f\ x|)$
by *force*

have *triange-inequality-main*: $\bigwedge i\ x.\ 1 \leq i \wedge i \leq N \longrightarrow |G\text{-}Nf\ x - f\ x| \leq I\text{-}1\ i\ x + I\text{-}2\ i\ x$
using *I-1-def I-2-def* **by** *force*

have *x-minus-xk-ge-h-on-Left-Half*:
$\forall k.\ k \in \{0..i-1\} \longrightarrow x - xs\ N\ !\ k \geq h$
proof (*clarify*)
fix *k*
assume *k-def*: $k \in \{0..i-1\}$
then have *k-pred-lt-i-pred*: $real\ k - 1 < real\ i - 1$
using *i-ge-1* **by** *fastforce*
have $x - xs\ N!k = x - (a + (real\ k - 1) * h)$
proof(*cases* $k=0$)
show $k = 0 \Longrightarrow x - xs\ N\ !\ k = x - (a + (real\ k - 1) * h)$
by (*simp add*: *zeroth-element*)
next
assume *k-nonzero*: $k \neq 0$
then have *k-def2*: $k \in \{1..N+1\}$
using *i-def k-def less-diff-conv2* **by** *auto*
then have $x - xs\ N\ !\ k = x - (a + (real\ k - 1) * h)$
by (*simp add*: *xs-els*)
then show *?thesis*
using *k-nonzero* **by** *force*
qed
also have ... $\geq h$
proof(*cases* $k=0$)
show $k = 0 \Longrightarrow h \leq x - (a + (real\ k - 1) * h)$
using *x-in-ab* **by** *force*
next
assume *k-nonzero*: $k \neq 0$
then have *k-type*: $k \in \{1..N\}$
using *i-leq-N k-def* **by** *fastforce*
have *difference-of-terms*: $(xs\ N!i) - (a+(real\ k - 1)*h) = ((real\ i-1) - (real\ k-1))*h$
by (*simp add*: *xs-i left-diff-distrib'*)
then have *first-inequality*: $x - (a + (real\ k - 1) * h) \geq (xs\ N!i) - (a+(real\ k - 1)*h)$
using *i-def* **by** *auto*
have *second-inequality*: $(xs\ N!i) - (a+(real\ k - 1)*h) \geq h$
using *difference-of-terms h-pos k-def k-nonzero* **by** *force*
then show *?thesis*
using *first-inequality* **by** *auto*
qed

finally show $h \leq x - xs\ N\ !\ k$**.**
qed

have *x-minus-xk-le-neg-h-on-Right-Half*:
$\forall k.\ k \in \{i+2..N+1\} \longrightarrow x - xs\ N\ !\ k \leq -h$
proof (*clarify*)
fix *k*
assume *k-def*: $k \in \{i+2..N+1\}$
then have *i-lt-k-pred*: $i < k-1$
by (*metis Suc-1 add-Suc-right atLeastAtMost-iff less-diff-conv less-eq-Suc-le*)
then have *k-nonzero*: $k \neq 0$
by *linarith*
from *i-lt-k-pred* **have** *i-minus-k-pred-leq-Minus-One*: $i - real\ (k - 1) \leq -1$
by *simp*
have $x - xs\ N!k = x - (a + (real\ k - 1) * h)$
proof−
have *k-def2*: $k \in \{1..N+1\}$
using *i-def k-def less-diff-conv2* **by** *auto*
then have $x - xs\ N\ !\ k = x - (a + (real\ k - 1) * h)$
using *xs-els* **by** *force*
then show *?thesis*
using *i-lt-k-pred* **by** *force*
qed
also have ... $\leq -h$
proof −
have *x-upper-limit*: $(xs\ N!(i+1)) = (a+(real\ i)*h)$
using *i-def xs-els* **by** *fastforce*
then have *difference-of-terms*: $(xs\ N!(i+1)) - (a+(real\ k - 1)*h) = ((real\ i) - (real\ k-1))*h$
by (*smt* (*verit*, *ccfv-threshold*) *diff-is-0-eq i-lt-k-pred left-diff-distrib′ nat-less-real-le nle-le of-nat-1 of-nat-diff of-nat-le-0-iff*)
then have *first-inequality*: $x - (a + (real\ k - 1) * h) \leq (xs\ N!(i+1)) - (a+(real\ k - 1)*h)$
using *i-def* **by** *fastforce*
have *second-inequality*: $(xs\ N!(i+1)) - (a+(real\ k - 1)*h) \leq -h$
by (*metis diff-is-0-eq′ difference-of-terms h-pos i-lt-k-pred i-minus-k-pred-leq-Minus-One linorder-not-le mult.left-commute mult.right-neutral mult-minus1-right nle-le not-less-zero of-nat-1 of-nat-diff ordered-comm-semiring-class.comm-mult-left-mono*)
then show *?thesis*
by (*smt* (*z3*) *combine-common-factor difference-of-terms first-inequality x-upper-limit*)
qed
finally show $x - xs\ N\ !\ k \leq -h$**.**
qed

have *I1-final-bound*: $I\text{-}1\ i\ x < (1+ (Sup\ ((\lambda x.\ |f\ x|)\ `\ \{a..b\}))) * \eta$

proof −

have *I1-decomp*:
$I\text{-}1\ i\ x \le (\sum k{\in}\{2..i{-}1\}.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|)$
$+ |f\ (a)| * |\sigma\ (w * (x - xs\ N\ !\ 0)) - 1|$
$+ (\sum k{\in}\{i{+}2..N{+}1\}.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
proof (*cases* $i < 3$)
assume *i-lt-3*: $i < 3$
then have *i-is-1-or-2*: $i = 1 \vee i = 2$
using *i-ge-1* **by** *linarith*
then have *empty-summation*:
$(\sum k = 2..i - 1.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|) = 0$
by *fastforce*
have *Lix*: $L\ i\ x = f(a) + (f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3)) + (f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2))$
using *L-def i-is-1-or-2* **by** *presburger*
have $I\text{-}1\ i\ x = |G\text{-}Nf\ x - L\ i\ x|$
by (*meson I-1-def i-ge-1 i-leq-N*)
also have ... $= |(\sum k{\in}\{2..N{+}1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$+ f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0))$
$- f(a)$
$- (f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3))$
$- (f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2))|$
by (*simp add*: *G-Nf-def Lix*)
also have ... $= |(\sum k{\in}\{3..N{+}1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$+ f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0))$
$- f(a)$
$- (f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3))|$
proof −
from *N-pos* **have** $(\sum k{\in}\{2..N{+}1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) =$
$(f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2)) +$
$(\sum k{\in}\{3..N{+}1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
by(*subst sum.atLeast-Suc-atMost*, *auto*)
then show *?thesis*
by *linarith*
qed

also have ... = $|(\sum k\in\{4..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$+ f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0))$
$- f(a)|$
proof −
from *N-gt-3* **have** $(\sum k\in\{3..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) =$
$(f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3)) +$
$(\sum k\in\{4..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
by(*subst sum.atLeast-Suc-atMost*, *simp-all*)
then show *?thesis*
by *linarith*
qed
also have ... = $|(\sum k\in\{4..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$+ f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
proof −
have $\forall real1\ real2\ real3.\ (real1::real) + real2 * real3 - real2 = real1 + real2 * (real3 - 1)$
by (*simp add*: *right-diff-distrib'*)
then show *?thesis*
using *first-element* **by** *presburger*
qed
also have ... $\leq |(\sum k\in\{4..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))|$
$+ |f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
by *linarith*
also have ... $\leq (\sum k\in\{4..N+1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))|)$
$+ |f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
using *add-mono* **by** *blast*
also have ... = $(\sum k\in\{4..N+1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
$+ |f\ (a)| * |(\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
by (*simp add*: *abs-mult*)
also have ... $\leq (\sum k\in\{i+2..N+1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
$+ |f\ (a)| * |(\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
proof(*cases* $i=1$)
assume *i-is-1*: $i = 1$
have *union*: $\{i+2\} \cup \{4..N+1\} = \{i+2..N+1\}$
proof(*safe*)

show $\bigwedge n.\ i + 2 \in \{i{+}2..N + 1\}$
using *N-gt-3 i-is-1* **by** *presburger*
show $\bigwedge n.\ n \in \{4..N + 1\} \Longrightarrow n \in \{i{+}2..N + 1\}$
using *i-is-1* **by** *auto*
show $\bigwedge n.\ n \in \{i{+}2..N + 1\} \Longrightarrow n \notin \{4..N + 1\} \Longrightarrow n \notin \{\} \Longrightarrow n = i + 2$
using *i-is-1* **by** *presburger*
qed
have $(\sum k\in\{4..N{+}1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
$+ |f\ (a)| * |(\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)| \leq$
$(\sum k\in\{i{+}2\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
$+$
$(\sum k\in\{4..N{+}1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
$+ |f\ (a)| * |(\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
by *auto*
also have ... $= (\sum k\in\{i{+}2..N{+}1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
$+ |f\ (a)| * |(\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
proof −
have $(\sum k\in\{i{+}2\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|) +$
$(\sum k\in\{4..N{+}1\}.\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|) =$
$(\sum k\in(\{i{+}2\} \cup \{4..N{+}1\}).\ |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| * |\sigma\ (w * (x - xs\ N\ !\ k))|)$
by (*subst sum.union-disjoint*, *simp-all*, *simp add*: *i-is-1*)
then show *?thesis*
using *union* **by** *presburger*
qed
finally show *?thesis*.
next
show $i \neq 1 \Longrightarrow$
$(\sum k = 4..N + 1.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k))|) + |f\ a| * |\sigma\ (w * (x - xs\ N\ !\ 0)) - 1|$
$\leq (\sum k = i + 2..N + 1.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k))|) + |f\ a| * |\sigma\ (w * (x - xs\ N\ !\ 0)) - 1|$
using *i-is-1-or-2* **by** *auto*
qed
finally show *?thesis*
using *empty-summation* **by** *linarith*
next
assume *main-case*: $\neg\ i < 3$
then have *three-leq-i*: $i \geq 3$

by *simp*
have *disjoint*: $\{2..i-1\} \cap \{i..N+1\} = \{\}$
by *auto*

have *union*: $\{2..i-1\} \cup \{i..N+1\} = \{2..N+1\}$
proof(*safe*)
show $\bigwedge n.\ n \in \{2..i - 1\} \Longrightarrow n \in \{2..N+1\}$
using *i-leq-N* **by** *force*
show $\bigwedge n.\ n\in \{i..N + 1\} \Longrightarrow n \in \{2..N + 1\}$
using *three-leq-i* **by** *force*
show $\bigwedge n.\ n \in \{2..N + 1\} \Longrightarrow n \notin \{i..N + 1\} \Longrightarrow n \in \{2..i - 1\}$
by (*metis Nat.le-diff-conv2 Suc-eq-plus1 atLeastAtMost-iff i-ge-1 not-less-eq-eq*)

qed

have *sum-of-terms*: $(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) +$
$(\sum k\in\{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) =$
$(\sum k\in\{2..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
using *sum.union-disjoint* **by** (*smt* (*verit*, *ccfv-threshold*) *disjoint union finite-atLeastAtMost*)

have $I\text{-}1\ i\ x = |G\text{-}Nf\ x - L\ i\ x|$
using *I-1-def i-ge-1 i-leq-N* **by** *presburger*
also have ... $= |G\text{-}Nf\ x - ((\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))) + f(a) +$
$(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i-1))) * \sigma\ (w * (x - xs\ N\ !\ i)) +$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ i)) * \sigma\ (w * (x - xs\ N\ !\ (i+1))))|$
by (*smt* (*verit*, *ccfv-SIG*) *main-case L-def less-add-one nat-1-add-1 numeral-Bit1 numeral-le-iff numerals*(*1*) *semiring-norm*(*70*) *three-leq-i*)
also have ... $= |(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) +$
$(\sum k\in\{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))\ + f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0)) -$
$(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))\ ) - f(a) - \ (f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i - 1)))* \sigma\ (w * (x - xs\ N\ !\ i)) -$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ i))* \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
by (*smt* (*verit*, *ccfv-SIG*) *G-Nf-def sum-mono sum-of-terms*)

also have ... $= |((\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$-(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))\ ))+$
$(\sum k\in\{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))\ + f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0))$

$- f(a) -$ $(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i - 1)))* \sigma\ (w * (x - xs\ N\ !\ i)) -$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ i))* \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
by *linarith*
also have ... $= |(\sum k \in \{2..i-1\}.$ $(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))$
$-(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))\ )+$
$(\sum k \in \{i..N+1\}.$ $(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$ $+ f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0))$
$- f(a) -$ $(f\ (xs\ N\ !\ (i)) - f\ (xs\ N\ !\ (i - 1)))* \sigma\ (w * (x - xs\ N\ !\ (i))) -$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ (i)))* \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
by (*simp add*: *sum-subtractf*)
also have ... $= |(\sum k \in \{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) +$
$(\sum k \in \{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) +$
$f\ (xs\ N\ !\ 1) * \sigma\ (w * (x - xs\ N\ !\ 0)) -$
$f(a) -$
$(f\ (xs\ N\ !\ (i)) - f\ (xs\ N\ !\ (i - 1))) * \sigma\ (w * (x - xs\ N\ !\ (i))) -$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ (i))) * \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
by (*simp add*: *right-diff-distrib′*)
also have ... $= |(\sum k \in \{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) +$
$(\sum k \in \{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) +$
$f\ (a) * \sigma\ (w * (x - xs\ N\ !\ 0)) -$
$f(a) -$
$(f\ (xs\ N\ !\ (i)) - f\ (xs\ N\ !\ (i - 1))) * \sigma\ (w * (x - xs\ N\ !\ (i))) -$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ (i))) * \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
using *first-element* **by** *fastforce*
also have ... $= |(\sum k \in \{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) +$
$(\sum k \in \{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) +$
$f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) -1)$
$-(f\ (xs\ N\ !\ (i)) - f\ (xs\ N\ !\ (i - 1))) * \sigma\ (w * (x - xs\ N\ !\ (i)))$
$-(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ (i))) * \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
by (*simp add*: *add-diff-eq right-diff-distrib′*)
also have ...$= |(\sum k \in \{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) +$
$f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1) +$
$(\sum k \in \{i+1..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
$- (f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ (i\ ))) * \sigma\ (w * (x - xs\ N\ !\ (i+1)))|$
proof −
from *i-leq-N* **have** $(\sum k \in \{i..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) =$
$(f\ (xs\ N\ !\ (i)) - f\ (xs\ N\ !\ (i - 1))) * \sigma\ (w * (x - xs\ N\ !\ (i))) +$
$(\sum k \in \{i+1..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x$

$-$ *xs N ! k*)))
by(*subst sum.atLeast-Suc-atMost*, *linarith*, *auto*)
then show *?thesis*
by *linarith*
qed
also have ...= $|(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) +$
$f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1) +$
$(\sum k\in\{i+2..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))|$
proof $-$
from *i-leq-N* **have** $(\sum k\in\{i+1..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) =$
$(f\ (xs\ N\ !\ (i+1)) - f\ (xs\ N\ !\ (i))) * \sigma\ (w * (x - xs\ N\ !\ (i+1))) +$
$(\sum k\in\{i+2..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))$
by(*subst sum.atLeast-Suc-atMost*, *linarith*, *auto*)
then show *?thesis*
by *linarith*
qed
show *?thesis*
proof $-$
have *inequality-pair*: $|\sum n = 2..i - 1.\ (f\ (xs\ N\ !\ n) - f\ (xs\ N\ !\ (n - 1))) * (\sigma\ (w * (x - xs\ N\ !\ n)) - 1)| \leq$
$(\sum n = 2..i - 1.\ |(f\ (xs\ N\ !\ n) - f\ (xs\ N\ !\ (n - 1))) * (\sigma\ (w * (x - xs\ N\ !\ n)) - 1)|) \wedge$
$|f\ a * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)| + |\sum n = i + 2..N + 1.\ (f\ (xs\ N\ !\ n) - f\ (xs\ N\ !\ (n - 1))) * \sigma\ (w * (x - xs\ N\ !\ n))|$
$\leq |f\ a * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)| + (\sum n = i + 2..N + 1.\ |(f\ (xs\ N\ !\ n) - f\ (xs\ N\ !\ (n - 1))) * \sigma\ (w * (x - xs\ N\ !\ n))|)$
using *add-le-cancel-left* **by** *blast*
have *I-1 i x* = $|(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) +$
$f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1) +$
$(\sum k = i + 2..N+1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))|$
using ‹$|(\sum k = 2..i - 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) + f\ a * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1) + (\sum k = i + 1..N + 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) - (f\ (xs\ N\ !\ (i + 1)) - f\ (xs\ N\ !\ i)) * \sigma\ (w * (x - xs\ N\ !\ (i + 1)))| = |(\sum k = 2..i - 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1)) + f\ a * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1) + (\sum k = i + 2..N + 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))|$›
calculation **by** *presburger*
also have ... $\leq|(\sum k\in\{2..i-1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * (\sigma\ (w * (x - xs\ N\ !\ k)) - 1))|$
$+ |f\ (a) * (\sigma\ (w * (x - xs\ N\ !\ 0)) - 1)|$
$+|(\sum k\in\{i+2..N+1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k)))|$

by *linarith*
also have ...≤ ($\sum$ k∈{2..i−1}. |(f (xs N ! k) − f (xs N ! (k − 1))) ∗ (σ (w ∗ (x − xs N ! k)) − 1)|)
+ |f (a) ∗ (σ (w ∗ (x − xs N ! 0)) − 1)|
+ ($\sum$ k∈{i+2..N+1}. |(f (xs N ! k) − f (xs N ! (k − 1))) ∗ σ (w ∗ (x − xs N ! k))|)
using *inequality-pair* **by** *linarith*
also have ...≤ ($\sum$ k∈{2..i−1}. |(f (xs N ! k) − f (xs N ! (k − 1)))| ∗ |(σ (w ∗ (x − xs N ! k)) − 1)|)
+ |f (a)| ∗ |σ (w ∗ (x − xs N ! 0)) − 1|
+ ($\sum$ k∈{i+2..N+1}. |(f (xs N ! k) − f (xs N ! (k − 1)))| ∗ |σ (w ∗ (x − xs N ! k))|)
proof −
have *f1*: ⋀k. k ∈ {2..i−1} ⟶ |(f (xs N ! k) − f (xs N ! (k − 1))) ∗ (σ (w ∗ (x − xs N ! k)) − 1)| ≤ |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k)) − 1|
by (*simp add*: *abs-mult*)
have *f2*: ⋀k. k ∈ {i+2..N+1} ⟶ |(f (xs N ! k) − f (xs N ! (k − 1))) ∗ σ (w ∗ (x − xs N ! k))| ≤ |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k))|
by (*simp add*: *abs-mult*)
have *f3*: |f (a) ∗ (σ (w ∗ (x − xs N ! 0)) − 1)| = |f (a)| ∗ |σ (w ∗ (x − xs N ! 0)) − 1|
using *abs-mult* **by** *blast*
then show *?thesis*
by (*smt* (*verit*, *best*) *f1 f2 sum-mono*)
qed
finally show *?thesis*.
qed
qed
also have ... < ($\sum$ k∈{2..i−1}. η ∗ (1/N)) +
|f (a)| ∗ |σ (w ∗ (x − xs N ! 0)) − 1| +
($\sum$ k∈{i+2..N+1}. η ∗ (1/N))
proof(*cases i* ≥ *3*)
assume *i-geq-3*: *3* ≤ *i*
show ($\sum$ k = 2..i − 1. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k)) − 1|) + |f a| ∗ |σ (w ∗ (x − xs N ! 0)) − 1| +
($\sum$ k = i + 2..N + 1. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k))|)
< ($\sum$ k = 2..i − 1. η ∗ (1 / N)) + |f a| ∗ |σ (w ∗ (x − xs N ! 0)) − 1| +
($\sum$ k = i + 2..N + 1. η ∗ (1 / N))
proof(*cases* ∀k. k ∈ {2..i−1} ⟶ |σ (w ∗ (x − xs N ! k)) − 1| = 0)
assume *all-terms-zero*: ∀k. k ∈ {2..i − 1} ⟶ |σ (w ∗ (x − xs N ! k)) − 1| = 0
from *i-geq-3* **have** ($\sum$ k∈{2..i−1}. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k)) − 1|) < ($\sum$ k∈{2..i−1}. η ∗ (1/N))
by (*subst sum-strict-mono*, *force*+, (*simp add*: *N-pos η-pos all-terms-zero*)+)
show *?thesis*

proof(*cases i = N*)
assume *i = N*
then show *?thesis*
using ‹(∑ *k = 2..i − 1. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k)) − 1|) < (∑ k = 2..i − 1. η ∗ (1 / N))*› **by** *auto*
next
assume *i ≠ N*
then have *i-lt-N*: *i < N*
using *i-leq-N le-neq-implies-less* **by** *blast*
show *?thesis*
proof(*cases ∀ k. k ∈ {i+2..N+1} ⟶ |σ (w ∗ (x − xs N ! k))| = 0*)
assume *all-second-terms-zero*: *∀ k. k ∈ {i + 2..N + 1} ⟶ |σ (w ∗ (x − xs N ! k))| = (0::real)*
from *i-lt-N* **have** *(∑ k∈{i+2..N+1}. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k))|) < (∑ k∈{i+2..N+1}. η ∗ (1/N))*
by(*subst sum-strict-mono*, *force+*, (*simp add*: *η-pos all-second-terms-zero*)+)
then show *?thesis*
proof −
show *?thesis*
using ‹(∑ *k = 2..i − 1. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k)) − 1|) < (∑ k = 2..i − 1. η ∗ (1 / N))*›
‹(∑ *k = i + 2..N + 1. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k))|) < (∑ k = i + 2..N + 1. η ∗ (1 / N))*› **by** *linarith*

qed
next
assume *second-terms-not-all-zero*: *¬ (∀ k. k ∈ {i + 2..N + 1} ⟶ |σ (w ∗ (x − xs N ! k))| = 0)*
obtain *NonZeroTerms* **where** *NonZeroTerms-def*: *NonZeroTerms = {k ∈ {i + 2..N + 1}. |σ (w ∗ (x − xs N ! k))| ≠ 0}*
by *blast*
obtain *ZeroTerms* **where** *ZeroTerms-def*: *ZeroTerms = {k ∈ {i + 2..N + 1}. |σ (w ∗ (x − xs N ! k))| = 0}*
by *blast*
have *zero-terms-eq-zero*: *(∑ k ∈ ZeroTerms. |f (xs N ! k) − f (xs N ! (k − 1))| ∗ |σ (w ∗ (x − xs N ! k))|) = 0*
by (*simp add*: *ZeroTerms-def*)
have *disjoint*: *ZeroTerms ∩ NonZeroTerms = {}*
using *NonZeroTerms-def ZeroTerms-def* **by** *blast*
have *union*: *ZeroTerms ∪ NonZeroTerms = {i+2..N+1}*
proof(*safe*)
show *⋀n. n ∈ ZeroTerms ⟹ n ∈ {i + 2..N + 1}*
using *ZeroTerms-def* **by** *force*
show *⋀n. n ∈ NonZeroTerms ⟹ n ∈ {i + 2..N + 1}*
using *NonZeroTerms-def* **by** *blast*
show *⋀n. n ∈ {i + 2..N + 1} ⟹ n ∉ NonZeroTerms ⟹ n ∈ ZeroTerms*
using *NonZeroTerms-def ZeroTerms-def* **by** *blast*
qed

have (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) <
(∑ *k*∈{*i*+*2*..*N*+*1*}. η ∗ ((*1*::*real*) / *real N*))
proof −
have (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) =
(∑ *k*∈*NonZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
proof −
have (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) =
(∑ *k*∈*ZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
+ (∑ *k*∈*NonZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
by (*smt disjoint finite-Un finite-atLeastAtMost union sum.union-disjoint*)
then show *?thesis*
using *zero-terms-eq-zero* **by** *linarith*
qed
also have ... < (∑ *k*∈*NonZeroTerms*. η ∗ (*1* / *N*))
proof(*rule sum-strict-mono*)
show *finite NonZeroTerms*
by (*metis finite-Un finite-atLeastAtMost union*)
show *NonZeroTerms* ≠ {}
using *NonZeroTerms-def second-terms-not-all-zero* **by** *blast*
fix *y*
assume *y-subtype*: *y* ∈ *NonZeroTerms*
then have *y-type*: *y* ∈ {*i*+*2*..*N*+*1*}
by (*metis Un-iff union*)
then have *y-suptype*: *y* ∈ {*1*..*N* + *1*}
by *simp*

have *parts-lt-eta*: ⋀*k*. *k*∈{*i*+*2*..*N*+*1*} ⟶ |(*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*)))| < η
proof(*clarify*)
fix *k*
assume *k-type*: *k* ∈ {*i* + *2*..*N* + *1*}
then have *k* − *1* ∈ {*i*+*1*..*N*}
by *force*
then have |(*xs N* ! *k*) − (*xs N* ! (*k* − *1*))| < δ ⟶ |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| < η
using *δ-prop atLeastAtMost-iff els-in-ab le-diff-conv* **by** *auto*

then show |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| < η
using *adj-terms-lt i-leq-N k-type* **by** *fastforce*
qed
then have *f-diff-lt-eta*: |*f* (*xs N* ! *y*) − *f* (*xs N* ! (*y* − *1*))| < η

using *y-type* **by** *blast*
have *lt-minus-h*: $x - xs\ N!y \leq -h$
using *x-minus-xk-le-neg-h-on-Right-Half y-type* **by** *blast*
then have *sigma-lt-inverseN*: $|\sigma\ (w * (x - xs\ N\ !\ y))| < 1\ /\ N$
proof −
have $\neg$ *Suc N* $< y$
using *y-suptype* **by** *force*
then show *?thesis*
by (*smt* (*z3*) *Suc-1 Suc-eq-plus1 lt-minus-h add.commute add.left-commute diff-zero length-map length-upt not-less-eq w-prop xs-eqs*)
qed

show $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| * |\sigma\ (w * (x - xs\ N\ !\ y))| < \eta * (1\ /\ N)$
using *f-diff-lt-eta mult-strict-mono sigma-lt-inverseN* **by** *fastforce*
qed
also have ... $\leq (\sum k{\in}NonZeroTerms.\ \eta * (1\ /\ N)) + (\sum k{\in}ZeroTerms.\ \eta * (1\ /\ N))$
using *η-pos* **by** *force*
also have ... $= (\sum k{\in}\{i{+}2..N{+}1\}.\ \eta * (1\ /\ N))$
by (*smt disjoint finite-Un finite-atLeastAtMost union sum.union-disjoint*)

finally show *?thesis*.
qed
then show *?thesis*
using ‹$(\sum k = 2..i - 1.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|) < (\sum k = 2..i - 1.\ \eta * (1\ /\ N))$› **by** *linarith*
qed
qed
next

assume *first-terms-not-all-zero*: $\neg\ (\forall k.\ k \in \{2..i - 1\} \longrightarrow |\sigma\ (w * (x - xs\ N\ !\ k)) - 1| = 0)$
obtain *BotNonZeroTerms* **where** *BotNonZeroTerms-def*: $BotNonZeroTerms = \{k \in \{2..i - 1\}.\ |\sigma\ (w * (x - xs\ N\ !\ k)) - 1| \neq 0\}$
by *blast*
obtain *BotZeroTerms* **where** *BotZeroTerms-def*: $BotZeroTerms = \{k \in \{2..i - 1\}.\ |\sigma\ (w * (x - xs\ N\ !\ k)) - 1| = 0\}$
by *blast*
have *bot-zero-terms-eq-zero*: $(\sum k \in BotZeroTerms.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|) = 0$
by (*simp add*: *BotZeroTerms-def*)
have *bot-disjoint*: $BotZeroTerms \cap BotNonZeroTerms = \{\}$
using *BotNonZeroTerms-def BotZeroTerms-def* **by** *blast*

have *bot-union*: $BotZeroTerms \cup BotNonZeroTerms = \{2..i - 1\}$
proof(*safe*)
show $\bigwedge n.\ n \in BotZeroTerms \Longrightarrow n \in \{2..i - 1\}$

using *BotZeroTerms-def* **by** *force*
show $\bigwedge n.\ n \in BotNonZeroTerms \Longrightarrow n \in \{2..i - 1\}$
using *BotNonZeroTerms-def* **by** *blast*
show $\bigwedge n.\ n \in \{2..i - 1\} \Longrightarrow n \notin BotNonZeroTerms \Longrightarrow n \in BotZeroTerms$
using *BotNonZeroTerms-def BotZeroTerms-def* **by** *blast*
qed

have $(\sum k\in\{2..i - 1\}.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) -1|) <$
$(\sum k\in\{2..i - 1\}.\ \eta * (1\ /\ N))$
proof −
have *disjoint-sum*: $sum\ (\lambda k.\ \eta * (1\ /\ N))\ BotNonZeroTerms + sum\ (\lambda k.\ \eta * (1\ /\ N))\ BotZeroTerms = sum\ (\lambda k.\ \eta * (1\ /\ N))\ \{2..i - 1\}$
proof −
from *bot-disjoint* **have** $sum\ (\lambda k.\ \eta * (1\ /\ real\ N))\ BotNonZeroTerms + sum\ (\lambda k.\ \eta * (1\ /\ N))\ BotZeroTerms =$
$sum\ (\lambda k.\ \eta * (1\ /\ real\ N))\ (BotNonZeroTerms \cup BotZeroTerms)$
by(*subst sum.union-disjoint*, (*metis*(*mono-tags*) *bot-union finite-Un finite-atLeastAtMost*)+, *auto*)
then show *?thesis*
by (*metis add.commute bot-disjoint bot-union finite-Un finite-atLeastAtMost sum.union-disjoint*)
qed

have $(\sum k\in\{2..i - 1\}.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|) =$
$(\sum k\in BotNonZeroTerms.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|)$
proof −
have $(\sum k\in\{2..i - 1\}.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) -1|) =$
$(\sum k\in BotZeroTerms.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) -1|)$
$+ (\sum k\in BotNonZeroTerms.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) -1|)$
by (*smt bot-disjoint finite-Un finite-atLeastAtMost bot-union sum.union-disjoint*)
then show *?thesis*
using *bot-zero-terms-eq-zero* **by** *linarith*
qed
also have ... $< (\sum k\in BotNonZeroTerms.\ \eta * (1\ /\ N))$
proof(*rule sum-strict-mono*)
show *finite BotNonZeroTerms*
by (*metis finite-Un finite-atLeastAtMost bot-union*)
show $BotNonZeroTerms \neq \{\}$
using *BotNonZeroTerms-def first-terms-not-all-zero* **by** *blast*
fix *y*

assume *y-subtype*: $y \in BotNonZeroTerms$
then have *y-type*: $y \in \{2..i - 1\}$
by (*metis Un-iff bot-union*)
then have *y-suptype*: $y \in \{1..N + 1\}$
using *i-leq-N* **by** *force*
have *parts-lt-eta*: $\bigwedge k.\ k \in \{2..i - 1\} \longrightarrow |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| < \eta$
proof(*clarify*)
fix k
assume *k-type*: $k \in \{2..i - 1\}$
then have $|(xs\ N\ !\ k) - (xs\ N\ !\ (k - 1))| < \delta \longrightarrow |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| < \eta$
by (*metis δ-prop add.commute add-le-imp-le-diff atLeastAtMost-iff diff-le-self dual-order.trans els-in-ab i-leq-N nat-1-add-1 trans-le-add2*)
then show $|f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| < \eta$
using *adj-terms-lt i-leq-N k-type* **by** *fastforce*
qed
then have *f-diff-lt-eta*: $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| < \eta$
using *y-type* **by** *blast*
have *lt-minus-h*: $x - xs\ N!y \geq h$
using *x-minus-xk-ge-h-on-Left-Half y-type* **by** *force*
then have *bot-sigma-lt-inverseN*: $|\sigma\ (w * (x - xs\ N\ !\ y)) - 1\ | < (1\ /\ N)$
by (*smt (z3) Suc-eq-plus1 add-2-eq-Suc′ atLeastAtMost-iff diff-zero length-map length-upt less-Suc-eq-le w-prop xs-eqs y-suptype*)
then show $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| * |\sigma\ (w * (x - xs\ N\ !\ y)) - 1| < \eta * (1\ /\ N)$
by (*smt (verit, del-insts) f-diff-lt-eta mult-strict-mono*)

qed
also have ... $\leq (\sum k \in BotNonZeroTerms.\ \eta * (1\ /\ N)) + (\sum k \in BotZeroTerms.\ \eta * (1\ /\ N))$
using *η-pos* **by** *force*
also have ... $= (\sum k \in \{2..i - 1\}.\ \eta * (1\ /\ N))$
using *sum.union-disjoint disjoint-sum* **by** *force*
finally show *?thesis*.
qed

show *?thesis*
proof(*cases* $i = N$)
assume $i = N$
then show *?thesis*
using ‹$(\sum k = 2..i - 1.\ |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| * |\sigma\ (w * (x - xs\ N\ !\ k)) - 1|) < (\sum k = 2..i - 1.\ \eta * (1\ /\ N))$› **by** *auto*
next
assume $i \neq N$
then have *i-lt-N*: $i < N$
using *i-leq-N le-neq-implies-less* **by** *blast*
show *?thesis*

proof(*cases* ∀ *k*. *k* ∈ {*i*+*2*..*N*+*1*} ⟶ |σ (*w* ∗ (*x* − *xs N* ! *k*))| = *0*)
assume *all-second-terms-zero*: ∀ *k*. *k* ∈ {*i* + *2*..*N* + *1*} ⟶ |σ (*w* ∗ (*x* − *xs N* ! *k*))| = *0*
from *i-lt-N* **have** (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) < (∑ *k*∈{*i*+*2*..*N*+*1*}. η ∗ (*1*/*N*))
by (*subst sum-strict-mono*, *fastforce*+, (*simp add*: *η-pos all-second-terms-zero*)+)
then show *?thesis*
using ‹(∑ *k* = *2*..*i* − *1*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*)) − *1*|) < (∑ *k* = *2*..*i* − *1*. η ∗ (*1* / *N*))› **by** *linarith*
next

assume *second-terms-not-all-zero*: ¬ (∀ *k*. *k* ∈ {*i* + *2*..*N* + *1*} ⟶ |σ (*w* ∗ (*x* − *xs N* ! *k*))| = *0*)
obtain *TopNonZeroTerms* **where** *TopNonZeroTerms-def*: *TopNonZeroTerms* = {*k* ∈ {*i* + *2*..*N* + *1*}. |σ (*w* ∗ (*x* − *xs N* ! *k*))| ≠ *0*}
by *blast*
obtain *TopZeroTerms* **where** *TopZeroTerms-def*: *TopZeroTerms* = {*k* ∈ {*i* + *2*..*N* + *1*}. |σ (*w* ∗ (*x* − *xs N* ! *k*))| = *0*}
by *blast*
have *zero-terms-eq-zero*: (∑ *k*∈*TopZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) = *0*
by (*simp add*: *TopZeroTerms-def*)
have *disjoint*: *TopZeroTerms* ∩ *TopNonZeroTerms* = {}
using *TopNonZeroTerms-def TopZeroTerms-def* **by** *blast*
have *union*: *TopZeroTerms* ∪ *TopNonZeroTerms* = {*i*+*2*..*N*+*1*}
proof(*safe*)
show ⋀*n*. *n* ∈ *TopZeroTerms* ⟹ *n* ∈ {*i* + *2*..*N* + *1*}
using *TopZeroTerms-def* **by** *force*
show ⋀*n*. *n* ∈ *TopNonZeroTerms* ⟹ *n* ∈ {*i* + *2*..*N* + *1*}
using *TopNonZeroTerms-def* **by** *blast*
show ⋀*n*. *n* ∈ {*i* + *2*..*N* + *1*} ⟹ *n* ∉ *TopNonZeroTerms* ⟹ *n* ∈ *TopZeroTerms*
using *TopNonZeroTerms-def TopZeroTerms-def* **by** *blast*
qed

have (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) <
(∑ *k*∈{*i*+*2*..*N*+*1*}. η ∗ (*1* / *N*))
proof −
have (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) =
(∑ *k*∈*TopNonZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
proof −
have (∑ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) =
(∑ *k*∈*TopZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
+ (∑ *k*∈*TopNonZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))|

$* |\sigma\ (w * (x - xs\ N\ !\ k))|)$
by (*smt disjoint finite-Un finite-atLeastAtMost union sum.union-disjoint*)
then show *?thesis*
using *zero-terms-eq-zero* **by** *linarith*
qed
also have ... $<$ ($\sum k\in TopNonZeroTerms.\ \eta * (1\ /\ N)$)
proof(*rule sum-strict-mono*)
show *finite TopNonZeroTerms*
by (*metis finite-Un finite-atLeastAtMost union*)
show *TopNonZeroTerms* $\neq \{\}$
using *TopNonZeroTerms-def second-terms-not-all-zero* **by** *blast*
fix *y*
assume *y-subtype*: $y \in TopNonZeroTerms$
then have *y-type*: $y \in \{i+2..N+1\}$
by (*metis Un-iff union*)
then have *y-suptype*: $y \in \{1..N + 1\}$
by *simp*
have *parts-lt-eta*: $\bigwedge k.\ k\in\{i+2..N+1\} \longrightarrow |(f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))| < \eta$
proof(*clarify*)
fix *k*
assume *k-type*: $k \in \{i + 2..N + 1\}$
then have $k - 1 \in \{i+1..N\}$
by *force*
then have $|(xs\ N\ !\ k) - (xs\ N\ !\ (k - 1))| < \delta \longrightarrow |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| < \eta$
using *δ-prop atLeastAtMost-iff els-in-ab le-diff-conv* **by** *auto*

then show $|f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| < \eta$
using *adj-terms-lt i-leq-N k-type* **by** *fastforce*
qed
then have *f-diff-lt-eta*: $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| < \eta$
using *y-type* **by** *blast*
have *lt-minus-h*: $x - xs\ N!y \le -h$
using *x-minus-xk-le-neg-h-on-Right-Half y-type* **by** *blast*
then have *sigma-lt-inverseN*: $|\sigma\ (w * (x - xs\ N\ !\ y))| < 1\ /\ N$
proof −
have $\neg\ Suc\ N < y$
using *y-suptype* **by** *force*
then show *?thesis*
by (*smt* (*z3*) *Suc-1 Suc-eq-plus1 lt-minus-h add.commute add.left-commute diff-zero length-map length-upt not-less-eq w-prop xs-eqs*)
qed
then show $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| * |\sigma\ (w * (x - xs\ N\ !\ y))| < \eta * (1\ /\ N)$
by (*smt* (*verit, best*) *f-diff-lt-eta mult-strict-mono*)
qed
also have ... $\le$ ($\sum k\in TopNonZeroTerms.\ \eta * (1\ /\ N)$) + ($\sum k\in TopZeroTerms.\ \eta * (1\ /\ N)$)

using *η-pos* **by** *force*
also have ... = ($\sum$ *k*∈{*i*+*2*..*N*+*1*}. *η* ∗ (*1* / *N*))
by (*smt disjoint finite-Un finite-atLeastAtMost union sum.union-disjoint*)

finally show *?thesis***.**
qed
then show *?thesis*
using ‹($\sum$ *k* = *2*..*i* − *1*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |*σ* (*w* ∗ (*x* − *xs N* ! *k*)) − *1*|) < ($\sum$ *k* = *2*..*i* − *1*. *η* ∗ (*1* / *N*))› **by** *linarith*
qed
qed
qed
next
assume ¬ *3* ≤ *i*
then have *i-leq-2*: *i* ≤ *2*
by *linarith*
then have *first-empty-sum*: ($\sum$ *k* = *2*..*i* − *1*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |*σ* (*w* ∗ (*x* − *xs N* ! *k*)) − *1*|) = *0*
by *force*
from *i-leq-2* **have** *second-empty-sum*: ($\sum$ *k* = *2*..*i* − *1*. *η* ∗ (*1* / *N*)) = *0*
by *force*
have *i-lt-N*: *i* < *N*
using *N-defining-properties i-leq-2* **by** *linarith*

have ($\sum$ *k* = *i* + *2*..*N* + *1*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |*σ* (*w* ∗ (*x* − *xs N* ! *k*))|) <
($\sum$ *k* = *i* + *2*..*N* + *1*. *η* ∗ (*1* / *N*))
proof(*cases* ∀ *k*. *k* ∈ {*i*+*2*..*N*+*1*} ⟶ |*σ* (*w* ∗ (*x* − *xs N* ! *k*))| = *0*)
assume *all-second-terms-zero*: ∀ *k*. *k* ∈ {*i* + *2*..*N* + *1*} ⟶ |*σ* (*w* ∗ (*x* − *xs N* ! *k*))| = *0*
from *i-lt-N* **have** ($\sum$ *k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |*σ* (*w* ∗ (*x* − *xs N* ! *k*))|) < ($\sum$ *k*∈{*i*+*2*..*N*+*1*}. *η* ∗ (*1*/*N*))
by (*subst sum-strict-mono*, *fastforce*+, (*simp add*: *η-pos all-second-terms-zero*)+)
then show *?thesis***.**
next
assume *second-terms-not-all-zero*: ¬ (∀ *k*. *k* ∈ {*i* + *2*..*N* + *1*} ⟶ |*σ* (*w* ∗ (*x* − *xs N* ! *k*))| = *0*)
obtain *NonZeroTerms* **where** *NonZeroTerms-def*: *NonZeroTerms* = {*k* ∈ {*i* + *2*..*N* + *1*}. |*σ* (*w* ∗ (*x* − *xs N* ! *k*))| ≠ *0*}
by *blast*
obtain *ZeroTerms* **where** *ZeroTerms-def*: *ZeroTerms* = {*k* ∈ {*i* + *2*..*N* + *1*}. |*σ* (*w* ∗ (*x* − *xs N* ! *k*))| = *0*}
by *blast*
have *zero-terms-eq-zero*: ($\sum$ *k*∈*ZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |*σ* (*w* ∗ (*x* − *xs N* ! *k*))|) = *0*
by (*simp add*: *ZeroTerms-def*)
have *disjoint*: *ZeroTerms* ∩ *NonZeroTerms* = {}
using *NonZeroTerms-def ZeroTerms-def* **by** *blast*

have *union*: *ZeroTerms* ∪ *NonZeroTerms* = {*i*+*2*..*N*+*1*}
proof(*safe*)
show $\bigwedge$*n*. *n* ∈ *ZeroTerms* ⟹ *n* ∈ {*i* + *2*..*N* + *1*}
using *ZeroTerms-def* **by** *force*
show $\bigwedge$*n*. *n* ∈ *NonZeroTerms* ⟹ *n* ∈ {*i* + *2*..*N* + *1*}
using *NonZeroTerms-def* **by** *blast*
show $\bigwedge$*n*. *n* ∈ {*i* + *2*..*N* + *1*} ⟹ *n* ∉ *NonZeroTerms* ⟹ *n* ∈ *ZeroTerms*
using *NonZeroTerms-def ZeroTerms-def* **by** *blast*
qed

have (∑*k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) <
(∑*k*∈{*i*+*2*..*N*+*1*}. η ∗ (*1* / *N*))
proof −
have (∑*k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) =
(∑*k*∈*NonZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
proof −
have (∑*k*∈{*i*+*2*..*N*+*1*}. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|) =
(∑*k*∈*ZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
+ (∑*k*∈*NonZeroTerms*. |*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*))| ∗ |σ (*w* ∗ (*x* − *xs N* ! *k*))|)
by (*smt disjoint finite-Un finite-atLeastAtMost union sum.union-disjoint*)
then show *?thesis*
using *zero-terms-eq-zero* **by** *linarith*
qed
also have ... < (∑*k*∈*NonZeroTerms*. η ∗ (*1* / *N*))
proof(*rule sum-strict-mono*)
show *finite NonZeroTerms*
by (*metis finite-Un finite-atLeastAtMost union*)
show *NonZeroTerms* ≠ {}
using *NonZeroTerms-def second-terms-not-all-zero* **by** *blast*
fix *y*
assume *y-subtype*: *y* ∈ *NonZeroTerms*
then have *y-type*: *y* ∈ {*i*+*2*..*N*+*1*}
by (*metis Un-iff union*)
then have *y-suptype*: *y* ∈ {*1*..*N* + *1*}
by *simp*

have *parts-lt-eta*: $\bigwedge$*k*. *k*∈{*i*+*2*..*N*+*1*} ⟶ |(*f* (*xs N* ! *k*) − *f* (*xs N* ! (*k* − *1*)))| < η
proof(*clarify*)
fix *k*

assume *k-type*: $k \in \{i + 2..N + 1\}$
then have $k - 1 \in \{i+1..N\}$
by *force*
then have $|(xs\ N\ !\ k) - (xs\ N\ !\ (k - 1))| < \delta \longrightarrow |f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| < \eta$
using *δ-prop atLeastAtMost-iff els-in-ab le-diff-conv* **by** *auto*

then show $|f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))| < \eta$
using *adj-terms-lt i-leq-N k-type* **by** *fastforce*
qed
then have *f-diff-lt-eta*: $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| < \eta$
using *y-type* **by** *blast*
have *lt-minus-h*: $x - xs\ N!y \leq -h$
using *x-minus-xk-le-neg-h-on-Right-Half y-type* **by** *blast*
then have *sigma-lt-inverseN*: $|\sigma\ (w * (x - xs\ N\ !\ y))| < 1\ /\ N$
proof −
have $\neg\ Suc\ N < y$
using *y-suptype* **by** *force*
then show *?thesis*
by (*smt* (*z3*) *Suc-1 Suc-eq-plus1 lt-minus-h add.commute add.left-commute diff-zero length-map length-upt not-less-eq w-prop xs-eqs*)
qed

show $|f\ (xs\ N\ !\ y) - f\ (xs\ N\ !\ (y - 1))| * |\sigma\ (w * (x - xs\ N\ !\ y))| < \eta * (1\ /\ N)$
using *f-diff-lt-eta mult-strict-mono sigma-lt-inverseN* **by** *fastforce*
qed
also have ... $\leq (\sum k \in NonZeroTerms.\ \eta * (1\ /\ N)) + (\sum k \in ZeroTerms.\ \eta * (1\ /\ N))$
using *η-pos* **by** *force*
also have ... $= (\sum k \in \{i+2..N+1\}.\ \eta * (1\ /\ N))$
by (*smt disjoint finite-Un finite-atLeastAtMost union sum.union-disjoint*)

finally show *?thesis***.**
qed
then show *?thesis***.**
qed
then show *?thesis*
using *first-empty-sum second-empty-sum* **by** *linarith*
qed

also have ... $= |f\ (a)| * |\sigma\ (w * (x - xs\ N\ !\ 0)) - 1| + (\sum k \in \{2..i-1\}.\ \eta * (1/N)) + (\sum k \in \{i+2..N+1\}.\ \eta * (1/N))$
by *simp*
also have ... $\leq |f\ (a)| * |\sigma\ (w * (x - xs\ N\ !\ 0)) - 1| + (\sum k \in \{2..N+1\}.\ \eta * (1/N))$
proof −
have $(\sum k \in \{2..i-1\}.\ \eta * (1/N)) + (\sum k \in \{i+2..N+1\}.\ \eta * (1/N)) \leq$

```
(∑k∈{2..N+1}. η ∗  (1/N))
        proof(cases i≥ 3)
          assume 3 ≤ i
          have disjoint: {2..i−1} ∩ {i+2..N+1} = {}
            by auto
          from i-leq-N have subset: {2..i−1} ∪ {i+2..N+1} ⊆ {2..N+1}
            by auto
           have sum-union: sum (λk. η ∗ (1 / N)) {2..i−1} + sum (λk. η ∗ (1 /
N)) {i+2..N+1} =
                          sum (λk. η ∗ (1 / N)) ({2..i−1} ∪ {i+2..N+1})
            by (metis disjoint finite-atLeastAtMost sum.union-disjoint)
         from subset η-pos have sum (λk. η ∗ (1 / N)) ({2..i−1} ∪ {i+2..N+1})
≤ sum (λk. η ∗ (1 / N)) {2..N+1}
            by(subst sum-mono2, simp-all)
          then show ?thesis
            using sum-union by auto
        next
          assume ¬ 3 ≤ i
          then have i-leq-2: i ≤ 2
            by linarith
          then have first-term-zero: (∑k = 2..i − 1. η ∗ (1 / N)) = 0
            by force
          from η-pos have (∑k = i + 2..N + 1. η ∗ (1 / N)) ≤ (∑k = 2..N +
1. η ∗ (1 / N))
            by(subst sum-mono2, simp-all)
          then show ?thesis
            using first-term-zero by linarith
        qed
        then show ?thesis
          by linarith
      qed
      also have ... = |f (a)| ∗ |σ (w ∗ (x − xs N ! 0)) − 1| + (N ∗ η ∗  (1/N))
      proof −
        have (∑k∈{2..N+1}. η ∗  (1/N)) = (N ∗ η ∗  (1/N))
          by(subst sum-constant, simp)
        then show ?thesis
          by presburger
      qed
      also have ... = |f (a)| ∗ |σ (w ∗ (x − xs N ! 0)) − 1| + η
        by (simp add: N-pos)
      also have ... ≤ |f (a)| ∗ (1/N) +  η
      proof −
        have |σ (w ∗ (x − xs N ! 0)) − 1| < 1/N
            by (smt (z3) Suc-eq-plus1-left ω-prop add-2-eq-Suc′ add-gr-0 atLeastAt-
Most-iff diff-zero
              length-map length-upt w-def x-in-ab xs-eqs zero-less-one zeroth-element)
        then show ?thesis
          by (smt (verit, ccfv-SIG) mult-less-cancel-left)
      qed
```

also have ... $\leq |f\ (a)| * \eta +\ \ \eta$
by (*smt* (*verit*, *best*) *mult-left-mono one-over-N-lt-eta*)
also have ... $=\ \ (1\ +\ \ |f\ (a)|) * \eta$
by (*simp add*: *distrib-right*)
also have ... $\leq (1+\ (SUP\ x \in \{a..b\}.\ |f\ x|)) * \eta$
proof −
from *a-lt-b* **have** $|f\ (a)| \leq (SUP\ x \in \{a..b\}.\ |f\ x|)$
by (*subst cSUP-upper*, *simp-all*, *metis bdd-above-Icc contin-f continuous-image-closed-interval continuous-on-rabs order-less-le*)
then show *?thesis*
by (*simp add*: *η-pos*)
qed
finally show *?thesis*.
qed

have *x-i-pred-minus-x-lt-delta*: $|xs\ N\ !(i-1) - x| < \delta$
proof −
have $|xs\ N\ !(i-1) - x| \leq |xs\ N\ !(i-1) - xs\ N!i| + |xs\ N!i - x|$
by *linarith*
also have ... $\leq 2*h$
proof −
have *first-inequality*: $|xs\ N\ !(i-1) - xs\ N!i| \leq h$
using *difference-of-adj-terms h-pos i-ge-1 i-leq-N* **by** *fastforce*
have *second-inequality*: $|xs\ N!i - x| \leq h$
by (*smt* (*verit*) *left-diff-distrib′ mult-cancel-right1 x-lower-bound-aux x-upper-bound-aux xs-Suc-i xs-i*)
show *?thesis*
using *first-inequality second-inequality* **by** *fastforce*
qed
also have ... $< \delta$
using *h-lt-δ-half* **by** *auto*
finally show *?thesis*.
qed
have *I2-final-bound*: $I\text{-}2\ i\ x < (2 * (Sup\ ((\lambda x.\ |\sigma\ x|)\ `\ UNIV)) + 1) * \eta$
proof(*cases* $i \geq 3$)
assume *three-lt-i*: $3 \leq i$
have *telescoping-sum*: $sum\ (\lambda k.\ f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k\ -\ 1)))\ \{2..i-1\} + f\ a = f\ (xs\ N\ !\ (i-1))$
proof(*cases* $i = 3$)
show $i = 3 \Longrightarrow (\sum k = 2..i\ -\ 1.\ f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k\ -\ 1))) + f\ a = f\ (xs\ N\ !\ (i\ -\ 1))$
using *first-element* **by** *force*
next
assume $i \neq 3$
then have *i-gt-3*: $i > 3$
by (*simp add*: *le-neq-implies-less three-lt-i*)
have $sum\ (\lambda k.\ f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k\ -\ 1)))\ \{2..i-1\} = f(xs\ N!(i-1)) - f(xs\ N!(2-1))$
proof −

have *f1*: $1 \le i - Suc\ 1$
using *three-lt-i* **by** *linarith*
have *index-shift*: $(\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ (k - 1))) = (\sum k \in \{1..i-2\}.\ f\ (xs\ N\ !\ k))$
by (*rule sum.reindex-bij-witness*[*of - λj. j +1 λj. j −1*], *simp-all*, *presburger*+)
have $sum\ (\lambda k.\ f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))\ \{2..i-1\} =$
$(\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ k)) - (\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ (k - 1)))$
by (*simp add*: *sum-subtractf*)
also have ... $= (\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ k)) - (\sum k \in \{1..i-2\}.\ f\ (xs\ N\ !\ k))$
using *index-shift* **by** *presburger*
also have ... $= (\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ k)) - (f\ (xs\ N\ !\ 1) + (\sum k \in \{2..i-2\}.\ f\ (xs\ N\ !\ k)))$
using *f1* **by** (*metis* (*no-types*) *Suc-1 sum.atLeast-Suc-atMost*)
also have ... $= ((\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ k)) - (\sum k \in \{2..i-2\}.\ f\ (xs\ N\ !\ k))) - f\ (xs\ N\ !\ 1)$
by *linarith*
also have ... $= (f\ (xs\ N\ !\ (i-1)) + (\sum k \in \{2..i-2\}.\ f\ (xs\ N\ !\ k)) - (\sum k \in \{2..i-2\}.\ f\ (xs\ N\ !\ k))) - f\ (xs\ N\ !\ 1)$
proof −
have *disjoint*: $\{2..i-2\} \cap \{i-1\} = \{\}$
by *force*
have *union*: $\{2..i-2\} \cup \{i-1\} = \{2..i-1\}$
proof(*safe*)
show $\bigwedge n.\ n \in \{2..i - 2\} \Longrightarrow n \in \{2..i - 1\}$
by *fastforce*
show $\bigwedge n.\ i - 1 \in \{2..i - 1\}$
using *three-lt-i* **by** *force*
show $\bigwedge n.\ n \in \{2..i - 1\} \Longrightarrow n \notin \{2..i - 2\} \Longrightarrow n \notin \{\} \Longrightarrow n = i - 1$
by *presburger*
qed
have $(\sum k \in \{2..i-2\}.\ f\ (xs\ N\ !\ k)) + f\ (xs\ N\ !\ (i-1)) = (\sum k \in \{2..i-2\}.\ f\ (xs\ N\ !\ k)) + (\sum k \in \{i-1\}.\ f\ (xs\ N\ !\ k))$
by *auto*
also have ... $= (\sum k \in \{2..i-2\} \cup \{i-1\}.\ f\ (xs\ N\ !\ k))$
using *disjoint* **by** *force*
also have ... $= (\sum k \in \{2..i-1\}.\ f\ (xs\ N\ !\ k))$
using *union* **by** *presburger*
finally show *?thesis*
by *linarith*
qed
also have ... $= f\ (xs\ N\ !\ (i-1)) - f\ (xs\ N\ !\ 1)$
by *auto*
finally show *?thesis*
by *simp*
qed
then show *?thesis*

using *first-element* **by** *auto*
qed

have *I2-decomp*: $I\text{-}2\ i\ x = |L\ i\ x - f\ x|$
using *I-2-def i-ge-1 i-leq-N* **by** *presburger*
also have ... $= |\ (((\sum k{\in}\{2..i{-}1\}.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1)))) + f(a)) +$
$(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i{-}1))) * \sigma\ (w * (x - xs\ N\ !\ i)) +$
$(f\ (xs\ N\ !\ (i{+}1)) - f\ (xs\ N\ !\ i)) * \sigma\ (w * (x - xs\ N\ !\ (i{+}1)))) - f\ x|$
using *L-def three-lt-i* **by** *auto*

also have ... $= |\ f\ (xs\ N\ !\ (i{-}1)) - f\ x +$
$(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i{-}1))) * \sigma\ (w * (x - xs\ N\ !\ i)) +$
$(f\ (xs\ N\ !\ (i{+}1)) - f\ (xs\ N\ !\ i)) * \sigma\ (w * (x - xs\ N\ !\ (i{+}1)))|$
using *telescoping-sum* **by** *fastforce*
also have ... $\leq |\ f\ (xs\ N\ !\ (i{-}1)) - f\ x| +$
$|(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i{-}1))) * \sigma\ (w * (x - xs\ N\ !\ i))| +$
$|(f\ (xs\ N\ !\ (i{+}1)) - f\ (xs\ N\ !\ i)) * \sigma\ (w * (x - xs\ N\ !\ (i{+}1)))|$
by *linarith*
also have ... $= |\ f\ (xs\ N\ !\ (i{-}1)) - f\ x| +$
$|(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i{-}1)))| * |\ \sigma\ (w * (x - xs\ N\ !\ i))| +$
$|(f\ (xs\ N\ !\ (i{+}1)) - f\ (xs\ N\ !\ i))| * |\sigma\ (w * (x - xs\ N\ !\ (i{+}1)))|$
by (*simp add*: *abs-mult*)
also have ... $<\ \eta +\ \eta * |\ \sigma\ (w * (x - xs\ N\ !\ i))| +\ \eta * |\sigma\ (w * (x - xs\ N\ !\ (i{+}1)))|$
proof −
from *x-in-ab x-i-pred-minus-x-lt-delta*
have *first-inequality*: $|f\ (xs\ N\ !\ (i{-}1)) - f\ x| < \eta$
by(*subst δ-prop*,
metis Suc-eq-plus1 add-0 add-le-imp-le-diff atLeastAtMost-iff els-in-ab i-leq-N less-imp-diff-less linorder-not-le numeral-3-eq-3 order-less-le three-lt-i,
simp-all)
from *els-in-ab i-leq-N le-diff-conv three-lt-i*
have *second-inequality*: $|(f\ (xs\ N\ !\ i) - f\ (xs\ N\ !\ (i{-}1)))| < \eta$
by(*subst δ-prop*,
simp-all,
metis One-nat-def add.commute atLeastAtMost-iff adj-terms-lt i-ge-1 trans-le-add2)
have *third-inequality*: $|(f\ (xs\ N\ !\ (i{+}1)) - f\ (xs\ N\ !\ i))| < \eta$
proof(*subst δ-prop*)
show $xs\ N\ !\ (i + 1) \in \{a..b\}$ **and** $xs\ N\ !\ i \in \{a..b\}$ **and** *True*
using *els-in-ab i-ge-1 i-leq-N* **by** *auto*
show $|xs\ N\ !\ (i + 1) - xs\ N\ !\ i| < \delta$
using *adj-terms-lt*
by (*metis Suc-eq-plus1 Suc-eq-plus1-left Suc-le-mono add-diff-cancel-left′ atLeastAtMost-iff i-leq-N le-add2*)
qed
then show *?thesis*

by (*smt* (*verit*, *best*) *first-inequality mult-right-mono second-inequality*)
qed
also have ... = (| σ ($w * (x - xs\ N\ !\ i)$)| + |σ ($w * (x - xs\ N\ !\ (i+1))$)| + *1*) $* \eta$
by (*simp add*: *mult.commute ring-class.ring-distribs*(*1*))
also have ... $\leq$ (*2*$*$ (*Sup* (($\lambda x.$ |σ x|) ‘ *UNIV*)) + *1*) $* \eta$
proof −
from *bounded-sigmoidal* **have** *first-inequality*: | σ ($w * (x - xs\ N\ !\ i)$)| $\leq$ (*Sup* (($\lambda x.$ |σ x|) ‘ *UNIV*))
by (*metis UNIV-I bounded-function-def cSUP-upper2 dual-order.refl*)

from *bounded-sigmoidal* **have** *second-inequality*: | σ ($w * (x - xs\ N\ !\ (i+1))$)| $\leq$ (*Sup* (($\lambda x.$ |σ x|) ‘ *UNIV*))
unfolding *bounded-function-def*
by (*subst cSUP-upper*, *simp-all*)
then show *?thesis*
using *η-pos first-inequality* **by** *auto*
qed
finally show *?thesis*.
next
assume $\neg$ *3* $\leq$ *i*
then have *i-is-1-or-2*: i = *1* $\vee$ i = *2*
using *i-ge-1* **by** *linarith*
have *x-near-a*: $|a - x| < \delta$
proof(*cases* i = *1*)
show i = *1* $\Longrightarrow$ $|a - x| < \delta$
using *first-element h-pos x-i-pred-minus-x-lt-delta x-lower-bound-aux zeroth-element* **by** *auto*
show $i \neq$ *1* $\Longrightarrow$ $|a - x| < \delta$
using *first-element i-is-1-or-2 x-i-pred-minus-x-lt-delta* **by** *auto*
qed

have *Lix*: $L\ i\ x = f(a) + (f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3)) + (f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2))$
using *L-def i-is-1-or-2* **by** *presburger*
have *I-2* $i\ x = |L\ i\ x - f\ x|$
using *I-2-def i-ge-1 i-leq-N* **by** *presburger*
also have ... = $|(f\ a - f\ x) + (f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3)) + (f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2))|$
using *Lix* **by** *linarith*
also have ... $\leq |(f\ a - f\ x)| + |(f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)) * \sigma\ (w * (x - xs\ N\ !\ 3))| + |(f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)) * \sigma\ (w * (x - xs\ N\ !\ 2))|$
by *linarith*
also have ... $\leq |(f\ a - f\ x)| + |f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)| * |\sigma\ (w * (x - xs\ N\ !\ 3))| + |f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)| * |\sigma\ (w * (x - xs\ N\ !\ 2))|$
by (*simp add*: *abs-mult*)
also have ... $< \eta + \eta * |\ \sigma\ (w * (x - xs\ N\ !\ 3))| + |f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)| * |\sigma\ (w * (x - xs\ N\ !\ 2))|$
proof −

from *x-in-ab x-near-a* **have** *first-inequality*: $|f\ a - f\ x| < \eta$
by(*subst δ-prop*, *auto*)
have *second-inequality*: $|f\ (xs\ N\ !\ 3) - f\ (xs\ N\ !\ 2)| < \eta$
proof(*subst δ-prop*, *safe*)
show $xs\ N\ !\ 3 \in \{a..b\}$
using *N-gt-3 els-in-ab* **by** *force*
show $xs\ N\ !\ 2 \in \{a..b\}$
using *N-gt-3 els-in-ab* **by** *force*
from *N-gt-3* **have** $xs\ N\ !\ 3 - xs\ N\ !\ 2 = h$
by(*subst xs-els*, *auto*, *smt* (*verit*, *best*) *h-pos i-is-1-or-2 mult-cancel-right1 nat-1-add-1 of-nat-1 of-nat-add xs-Suc-i xs-i*)
then show $|xs\ N\ !\ 3 - xs\ N\ !\ 2| < \delta$
using *adj-terms-lt first-element zeroth-element* **by** *fastforce*
qed
then show *?thesis*
by (*smt* (*verit*, *best*) *first-inequality mult-right-mono*)
qed
also have ... $\leq\ \eta +\ \eta * |\ \sigma\ (w * (x - xs\ N\ !\ 3))| + \eta * |\sigma\ (w * (x - xs\ N\ !\ 2))|$
proof −
have *third-inequality*: $|f\ (xs\ N\ !\ 2) - f\ (xs\ N\ !\ 1)| < \eta$
proof(*subst δ-prop*, *safe*)
show $xs\ N\ !\ 2 \in \{a..b\}$
using *N-gt-3 els-in-ab* **by** *force*
show $xs\ N\ !\ 1 \in \{a..b\}$
using *N-gt-3 els-in-ab* **by** *force*
from *N-pos first-element* **have** $xs\ N\ !\ 2 - xs\ N\ !\ 1 = h$
by(*subst xs-els*, *auto*)
then show $|xs\ N\ !\ 2 - xs\ N\ !\ 1| < \delta$
using *adj-terms-lt first-element zeroth-element* **by** *fastforce*
qed
show *?thesis*
by (*smt* (*verit*, *best*) *mult-right-mono third-inequality*)
qed
also have ... $= (|\ \sigma\ (w * (x - xs\ N\ !\ 3))| + |\sigma\ (w * (x - xs\ N\ !\ 2))| + 1)*\eta$
by (*simp add*: *mult.commute ring-class.ring-distribs*(*1*))
also have ... $\leq (2*(Sup\ ((\lambda x.\ |\sigma\ x|)\ `\ UNIV)) + 1) * \eta$
proof −
from *bounded-sigmoidal* **have** *first-inequality*: $|\ \sigma\ (w * (x - xs\ N\ !\ 3))| \leq Sup\ ((\lambda x.\ |\sigma\ x|)\ `\ UNIV)$
unfolding *bounded-function-def*
by (*subst cSUP-upper*, *simp-all*)
from *bounded-sigmoidal* **have** *second-inequality*: $|\ \sigma\ (w * (x - xs\ N\ !\ 2))| \leq Sup\ ((\lambda x.\ |\sigma\ x|)\ `\ UNIV)$
unfolding *bounded-function-def*
by (*subst cSUP-upper*, *simp-all*)
then show *?thesis*
using *η-pos first-inequality* **by** *force*
qed

finally show *?thesis*.
qed

have $|(\sum k = 2..N + 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) + f\ a * \sigma\ (w * (x - xs\ N\ !\ 0)) - f\ x| \leq I\text{-}1\ i\ x + I\text{-}2\ i\ x$
using *G-Nf-def i-ge-1 i-leq-N triange-inequality-main first-element* **by** *blast*
also have $... < (1 + (Sup\ ((\lambda x.\ |f\ x|)\ `\ \{a..b\}))) * \eta + (2 * (Sup\ ((\lambda x.\ |\sigma\ x|)\ `\ UNIV)) + 1) * \eta$
using *I1-final-bound I2-final-bound* **by** *linarith*
also have $... = ((Sup\ ((\lambda x.\ |f\ x|)\ `\ \{a..b\})) + 2*(Sup\ ((\lambda x.\ |\sigma\ x|)\ `\ UNIV)) + 2) * \eta$
by (*simp add*: *distrib-right*)
also have $... = \varepsilon$
using *η-def η-pos* **by** *force*
finally show $|(\sum k = 2..N + 1.\ (f\ (xs\ N\ !\ k) - f\ (xs\ N\ !\ (k - 1))) * \sigma\ (w * (x - xs\ N\ !\ k))) + f\ a * \sigma\ (w * (x - xs\ N\ !\ 0)) - f\ x| < \varepsilon$.
qed
qed

end
theory *Sigmoid-Universal-Approximation*
imports *Limits-Higher-Order-Derivatives*
Sigmoid-Definition
Derivative-Identities-Smoothness
Asymptotic-Qualitative-Properties
Universal-Approximation
begin

end

References

[1] T. Chen, H. Chen, and R.-w. Liu. A constructive proof and an extension of Cybenko's approximation theorem. In C. Page and R. LePage, editors, *Computing Science and Statistics*, pages 163–168, New York, NY, 1992. Springer New York.

[2] D. Costarelli and R. Spigler. Constructive approximation by superposition of sigmoidal functions. *Analysis in Theory and Applications*, 29:169–196, 06 2013.

[3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control Signal Systems 2, 303314*, https://doi.org/10.1007/BF02551274, 1989.

[4] A. A. Minai and R. D. Williams. On the derivatives of the sigmoid. *Neural Networks*, 6(6):845–853, 1993.

[5] T. Nipkow, M. Wenzel, and L. C. Paulson. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, 2002.