Chapter 9

Affine Functions: Local Analysis

Affine functions are the result of adding-on a *constant function* to a *dilation function* (or the other way around). In other words, affine functions are specified by a *global input-output rule* of the form

$$
\begin{aligned}
x \xrightarrow{\quad AFFINE_{a,b} \quad} AFFINE_{a,b}(x) &= ax^{+1} + bx^{0} \\
&= ax + b
\end{aligned}
$$

where $AFFINE_{a,b}$ is the name of the function[1] and where a and b stand for the two *bounded* numbers needed to specify the function $AFFINE_{a,b}$.

EXAMPLE 1. The function specified by the global input-output rule

$$
\begin{aligned}
x \xrightarrow{\quad NINA_{(-3,+5)} \quad} NINA_{-3,+5}(x) &= (-3)x^{+1} + (+5) \\
&= -3x + 5
\end{aligned}
$$

is an affine function.

[1] Educologists seem to have remained blissully unaware that, some over fifty years ago, "linear" came to take on a different meaning, as in e.g. LINEAR ALGEBRA, so that "linear functions" came to be known as "affine functions".

linear term
linear coefficient
constant term
constant coefficient

In order to facilitate our investigation of affine functions, it is necessary to introduce some language special to affine functions.

Given the affine function $AFFINE_{a,b}$, that is the function specified by the global input-ouput rule

$$x \xrightarrow{AFFINE} AFFINE(x) = ax + b$$

- The term ax, AKA ax^{+1}, which is the term that corresponds to the dilation function is called the **linear term** of the affine function $AFFINE_{a,b}$ and the coefficient a is called the **linear coefficient** of the affine function $AFFINE_{a,b}$.
- The term b, AKA bx^0, which is the term that corresponds to the constant function is called the **constant term** of the affine function $AFFINE_{a,b}$ and the coefficient b is called the **constant coefficient** of the affine function $AFFINE_{a,b}$. Thus, depending on our viewpoint, b is the constant term or the constant coefficient. [2].

EXAMPLE 2. Given the affine function $NINA_{-3,+5}$, that is the function specified by the input-output rule

$$\begin{aligned} x \xrightarrow{NINA} NINA(x) &= (-3)x^{+1} + (+5) \\ &= -3x + 5 \end{aligned}$$

we have that

- $-3x$ is the linear term and -3 is the linear coefficient,
- $+5$ is the constant term and also the constant coefficient.

9.1 Input-Output Pairs

In the case of an affine function $AFFINE_{a,b}$, that is of a function whose global input-output rule is

$$x \xrightarrow{AFFINE} AFFINE(x) = ax + b$$

and given an input x_0, in order to get the output $AFFINE(x_0)$, we proceed as follows:

i. We *read* the global input-output rule:

> The output of $AFFINE_{a,b}$ is obtained by *multiplying* a by (1 copy of) the input and adding b

[2] Educologists will of course deplore this "hair splitting" but, had we not distinguished the constant *term* from the constant *coefficient*, they would have been the first to deplore the resulting confusion. At least one hopes so.

ii. We indicate that x is to be replaced by the given input x_0

$$x\Big|_{x\leftarrow x_0} \xrightarrow{AFFINE} AFFINE(x)\Big|_{x\leftarrow x_0} = ax+b\Big|_{x\leftarrow x_0}$$

iii. When we are given specific numbers for a, b, and x_0, we can then *identify* the resulting specifying-phrase ax_0+b to get the output $AFFINE(x_0)$.

EXAMPLE 3. Given the affine function specified by the global input-output rule

$$x \xrightarrow{ALDA} ALDA(x) = -32.67x + 71.07$$

and given the input -3, in order to get the input-output pair, we proceed as follows.

i. We *read* the global input-output rule:

The output of $ALDA$ is obtained by *multiplying* -32.67 by 1 copy of the input and adding 71.07

ii. We indicate that x is to be replaced by the given input -3

$$x\Big|_{x\leftarrow -3} \xrightarrow{ALDA} ALDA(x)\Big|_{x\leftarrow -3} = -32.67x + 71.07\Big|_{x\leftarrow -3}$$

iii. We *identify* the resulting specifying-phrase to get the output.

$$\begin{aligned} &= (-32.67)\cdot(-3) + 71.07 \\ &= +98.01 + 71.07 \\ &= +169.08 \end{aligned}$$

iv. Altogether, depending on the circumstances, we can then write any of the following:

$$-3 \xrightarrow{ALDA} +169.08$$

$$ALDA(-3) = +169.08$$

$(-3, +169.08)$ is an *input-output pair* for $ALDA$

.

LOCAL ANALYSIS NEAR INFINITY

Doing local analysis means working in a neighborhood of some given input and thus counting inputs from the given input since it is the center of the neighborhood. When the given input is ∞, counting from ∞ means to set $x \leftarrow$ *large* and to compute with powers of *large* in descending order of sizes.

NOTE. As it happens, the powers in global input-output rules are usually given in descending order of exponents which, when x is *large*, is in descending order of sizes. However, this is not an obligation and, when the powers in a global input-output rule are not given in descending order of exponents, in order to work in a neighborhood of ∞, the powers *must* be rearranged in descending order of exponents so as to be in descending order of sizes.

principal term near ∞
+[...]
approximate local input-output rule

9.2 Local Input-Output Rule Near Infinity

Given an affine function $AFFINE$, we look for a function whose input-output rule will be simpler than the input-output rule of $AFFINE$ but whose local graph near ∞ will still be qualitatively the same as the local graph near ∞ of $AFFINE$.

1. Given an affine function $AFFINE_{a,b}$ that is a function specified by the *global* input-ouput rule

$$x \xrightarrow{AFFINE_{a,b}} AFFINE_{a,b}(x) = ax + b$$

where a and b are *bounded*, when the input is *large*, the linear term of $AFFINE_{a,b}$ is the **principal term near** ∞ of the output, because when the input is near ∞:

- The *bounded* number a multiplied by a (copy of a) *large* input gives a result that is *large*,

and

- Adding a *bounded* number, regardless of its *sign*, to a *large* number will *not* change the *size* of the result which which will remain *large*.

2. However even though, for *large* inputs, the *constant term* of $AFFINE$ is "too small to matter here", it is not 0 and so we cannot write that $AFFINE_{a,b}(x)$ is equal to ax and, in mathematics, we do want to write *equalities*, if only because they are easier to work with. So we will use the symbol +[...] to mean "plus something too small to matter *here*" and we will write

$$x|_{x \text{ near } \infty} \xrightarrow{AFFINE_{a,b}} AFFINE_{a,b}(x)|_{x \text{ near } \infty} = ax + [...]$$

which we will of course call the **approximate local input-output rule** near ∞ of the affine function $AFFINE_{a,b}$[3]

Example 4. Given the function specified by the global input-output rule

$$x \xrightarrow{ALMA} ALMA(x) = -71.05x - 41.83$$

we can write

$$x|_{x \text{ near } \infty} \xrightarrow{JOAN} ALMA(x)|_{x \text{ near } \infty} = -71.05x + [...]$$

[3]Educologist who wondered why we used the term "global input-output rule" rather than just "input-ouput rule" will now see that it was to emphasize the difference between *global* input-output rules which work for *any* input and a *local* input-output rules which work only for inputs *in a given neighborhood.*

NOTE. This is exactly the same as in ARITHMETIC where $\frac{1}{3}$ is not equal to 0.33 because $3 \times \frac{1}{3} = 1$ while $3 \times 0.33 = 0.99$ so that we can only write $\frac{1}{3} = 0.33 + [...]$.

In other words, we have:

THEOREM 1 (Approximation Near ∞). *Near ∞, the output of an affine function is principally its* linear term.

9.3 Local Graphs Near Infinity

We get the *qualitative* local graph near ∞ as follows:

i. We use the **Approximation Near ∞ Theorem** to get the *approximate local input-output rule near ∞*
ii. We normalize it so that, depending on the sign of the coefficient, we get the local input-output rule of either the *identity function* or the *opposite function*,
iii. We draw the corresponding *local graph near ∞*..

EXAMPLE 5. Given the affine function $ALIX_{-21.36,-45.78}$ that is the function specified by the *global input-output rule*

$$x \xrightarrow{ALIX} ALIX(x) = -21.36x - 45.78$$

find its local graph near ∞.

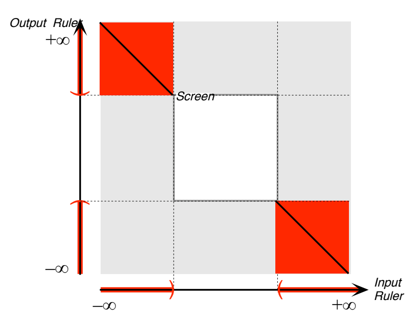
i. The approximate *local input-output rule* near ∞ is:

$$x|_{x \text{ near } \infty} \xrightarrow{ALIX} ALIX(x)|_{x \text{ near } \infty} = -21.36x^{+1} + [...]$$

ii. We *normalize* $ALIX$

$$x|_{x \text{ near } \infty} \xrightarrow{ALIX} ALIX(x)|_{x \text{ near } \infty} = -x^{+1} + [...]$$

iii. The *local graph* near ∞ of $ALIX$ is:



9.4 Local Features Near Infinity

Once we have the local graph of the function $AFFINE$ near ∞, we can "read" the local features off the local graph near ∞ as in **Chapter 4**. This is a good way to proceed but not in the long run because it is too slow. And, in fact, once we have a mental picture of the two possible qualitative local graphs near ∞, in order to decide which of the two is the actual case in the instance, we need only find the *place* of the local graph which is given by

THEOREM 2 (Height-sign Near ∞). *Given the function $AFFINE_{a,b}$:*

- $When$ Sign $a = +,$ Height-sign $AFFINE\,|_{\text{near } \infty} = (+, -)$
- $When$ Sign $a = -,$ Height-sign $AFFINE\,|_{\text{near } \infty} = (-, +)$

(Keep in mind that the feature signs near ∞ is coded while facing ∞ and not while facing 0.)
A more compact way of stating the above theorem is:

THEOREM 2 (Height-sign Near ∞). *Given the function $AFFINE_{a,b}$:*

$$\text{Height-sign } AFFINE\,|_{\text{near } \infty} = (\text{Sign } a, \, -\text{Sign } a)$$

And of course we also have the other local feature near ∞:

THEOREM 3 (Slope-sign Near ∞). *Given the function $AFFINE_{a,b}$:*

- *When Sign* $a = +,$ Slope-sign $AFFINE\,|_{near\ x_0} = (\diagup, \diagup)$
- *When Sign* $a = -,$ Slope-sign $AFFINE\,|_{near\ x_0} = (\diagdown, \diagdown)$

(Keep in mind that the feature signs near ∞ are coded while facing ∞ and not while facing 0.)

EXAMPLE 6. Given the affine function $MEOW_{-40.38,-94.21}$ that is the function specified by the global input-output rule

$$x \xrightarrow{MEOW} MEOW(x) = -40.38x - 94.21$$

we find its local features near ∞ as follows

i. The local features near ∞ according to the **Local Feature-sign Theorems** are:

- Height-sign $|_{x \text{ near } \infty} = (-, +)$
- Slope-sign $|_{x \text{ near } \infty} = (-, -)$

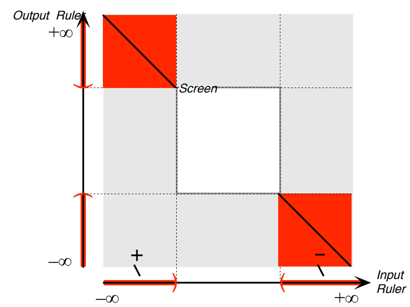
which, translated back into our language, gives

- Height-sign $|_{x \text{ near } \infty} = (+, +)$
- Slope-sign $|_{x \text{ near } \infty} = (\diagdown, \diagdown)$

ii. Should we want to check that the **Local Feature-signs Theorems** gave us the correct information, we would get the approximate local input-output rule near ∞

$$x|_{x \text{ near } \infty} \xrightarrow{MEOW} MEOW(x)|_{x \text{ near } \infty} = +x^2 + [...]$$

and then the local graph near ∞ from which we get the following local features

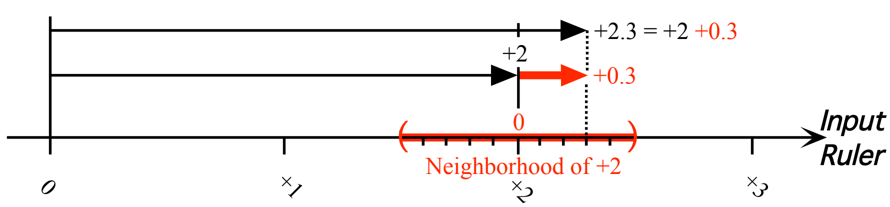


which are the same as those that were given by the **Feature-sign Near ∞ Theorems** keeping in mind that the local graph near ∞ is viewed here by **Mercator** rather than by **Magellan**.

Local Analysis near a Finite Input

Doing local analysis means working in a neighborhood of some given input and thus counting inputs from the given input since it is the *center* of the neighborhood. When the given input is x_0, we *localize* at x_0, that is we set $x = x_0 + h$ where h is *small* and we compute with powers of h in descending order of sizes. (See **Section 2.7 Analysis Near Finite Inputs**.

Example 7. Given the input $+2$, then the location of the number $+2.3$ relative to $+2$ is $+0.3$:



9.5 Local Input-Output Rule Near A Finite Input

Since x_0 is given for the duration of the local investigation and since it is therefore h that will be the actual input, we will think in terms of a **local**

local function

function, that is of a function which returns for h the same output that the given global function would return for $x = x_0 + h$.

In order to get the input-output rule of the *local function* function when the given function is an *affine function*, we proceed as follows:

> **i.** We replace x by $x_0 + h$ (where h is *small*) in the global input-output rule,
> **ii.** We use the **Multiplication Theorem** to compute with h.

More precisely, given the affine function $AFFINE_{a,b}$, that is the function specified by the global input-output rule

$$x \xrightarrow{AFFINE_{a,b}} AFFINE_{a,b}(x) = ax + b$$

and given an input x_0

i. We *localize* the function $AFFINE$ at the given input x_0, that is we count inputs from the given input x_0 instead of from the origin of the input ruler. In other words, we use the *location* of x in relation to the given input x_0, that is we replace in the global input-output rule x by $x_0 + h$, (where h is *small*).

$$\begin{aligned} x|_{x \leftarrow x_0+h} \xrightarrow{AFFINE_{a,b}} AFFINE_{a,b}(x)|_{x \leftarrow x_0+h} &= ax + b|_{x \leftarrow x_0+h} \\ &= a[x_0 + h] + b \\ &= ax_0 + ah + b \end{aligned}$$

ii. We collect terms of like size in order of diminishing sizes:

$$= [ax_0 + b] + ah$$

We then have:

THEOREM 4 (Local Input-Output Rule Near x_0). *Given the affine function $AFFINE$ specified by the* global input-ouput rule

$$x \xrightarrow{AFFINE} AFFINE(x) = ax + b$$

the local function $AFFINE_{(x_0)}$ is specified by the local input-output rule

$$h \xrightarrow{AFFINE_{(x_0)}} AFFINE_{(x_0)}(h) = \Big[ax_0 + b\Big] + \Big[a\Big]h$$

EXAMPLE 8. Given the affine function $MARA_{+3,+17}$, that is given the function whose global input-output rule is

local constant coefficient
local linear coefficient

$$x \xrightarrow{MARA} MARA(x) = +3x + 17$$

and given the input, -5, we *localize* the function $MARA$ at -5 as follows:

$$\begin{aligned} x|_{x=-5+h} \xrightarrow{\quad MARA \quad} MARA(x)|_{x\leftarrow -5+h} &= +3x + 17|_{x\leftarrow -5+h} \\ &= +3(-5+h) + 17 \\ &= -15 + 3h + 17 \end{aligned}$$

We collect terms of like size in order of diminishing sizes:

$$= +2 + 3h$$

The local function $MARA_{(-5)}$ is therefore specified by the *local input-output rule*

$$h \xrightarrow{MARA_{(-5)}} MARA_{(-5)}(h) = +2 + 3h$$

9.6 Local Coefficients Near A Finite Input

When looking only for *one* of the local features, instead of computing the whole local input-ouput rule, we will only compute the single term of the local input-ouput rule that controls the local feature.

More precisely, given the affine function $AFFINE_{a,b}$, that is a function specified by the global input-output rule

$$x \xrightarrow{AFFINE_{a,b}} AFFINE_{a,b}(x) = ax + b$$

and whose local input-output rule is therefore

$$h \xrightarrow{AFFINE_{(x_0)}} AFFINE_{(x_0)}(h) = \Big[ax_0 + b\Big] + \Big[a\Big]h$$

we will say that:

- $\Big[ax_0 + b\Big]$ is the **local constant coefficient**,
- $\Big[a\Big]$ is the **local linear coefficient**,

and we will now investigate how to get just a single one of the local coefficients without getting the whole local input-output rule. At first, getting just this one single term rather than the whold local input-output rule will look more difficult than getting the whole local input-output rule but, with a little bit of practice, writing less and less each time, this will soon get easy.

We begin by writing

$$\begin{aligned} x \mid_{x \leftarrow x_0+h} \xrightarrow{AFFINE_{a,b}} AFFINE_{a,b}(x) \mid_{x \leftarrow x_0+h} &= ax + b \mid_{x \leftarrow x_0+h} \\ &= a(x_0 + h) + b \end{aligned}$$

a. To find the *local constant coefficient*, we proceed as follows:
i. The contribution of $a(x_0 + h)$ to the *local constant term* will be $a \cdot x_0$
ii. The contribution of b to the local constant term will be b
Altogether, the *local constant term* adds up to

$$\Big[ax_0 + b\Big]$$

b. To find the *local linear coefficient*, we proceed as follows
i. The contribution of $a(x_0 + h)$ to the *local linear term* will be $a \cdot h$
ii. The contribution of b to the *local linear term* will be nothing.
Altogether, the *local linear term* adds up to

$$\Big[a\Big]h$$

Admittedly, here, the economy is not exactly staggering. On the other hand, though, this will serve to get us used to being able to compute each local coefficient independently of the others since, as we continue with our investigation of polynomial functions, any other way will eventually become ultra complicated.

9.7 Local Graph Near A Finite Input

The local graph of a given affine function $AFFINE_{a,b}$ near a given input x_0 is the graph of the *local function* $AFFINE_{(x_0)}$, that is the graph of the function specified by the *local input-output rule* of $AFFINE_{a,b}$ near x_0.

So, in order to get the local graph of the given affine function $AFFINE$ near the given input x_0, we:

i. Get the local graph of the *constant term* of the local rule $AFFINE_{(x_0)}$,

ii. Get the local graph of the *linear term* of the local rule $AFFINE_{(x_0)}$,

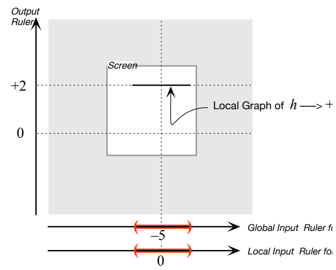
iii. Add-on (as we did in **Chapter 8**) the local graph of the *linear term* to the local graph of the *constant term* to get the local graph of $AFFINE_{(x_0)}$.

EXAMPLE 9. Given the affine function $MARA$ whose local rule near -5, $MARA_{(-5)}$ is given by the local input-output rule

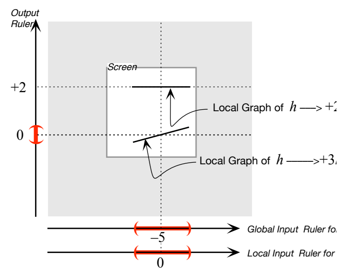
$$h \xrightarrow{MARA_{(-5)}} MARA_{(-5)} = +2 + 3h$$

find the local graph near -5

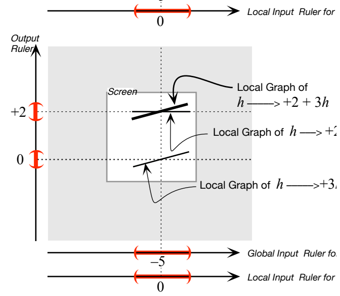
i. We get the local graph of the *constant term* of the local rule $MARA_{(-5)}$:



ii. We get the local graph near x_0 of the *linear term* of the local rule $MARA_{(-5)}$,



iii. We "add" the local graph of the *linear term* of the local rule $MARA_{(-5)}$ to the local graph of the *constant term* of $MARA_{(-5)}$.



9.8 Local Features Near A Finite Input

Once we have the local graph of the function $AFFINE$ near x_0, we can "read" the local features off the local graph near x_0 as in **Chapter 4**. This is a good way to proceed but not in the long run because it is too slow.

Given the affine function $AFFINE_{a,b}$, that is the function specified by the global input-output rule

$$x \xrightarrow{AFFINE} AFFINE(x) = ax + b$$

what *controls* the local features of $AFFINE$ near x_0 are the local coefficients, that is the coefficients in the local rule near x_0

$$h \xrightarrow{AFFINE_{(x_0)}} AFFINE_{(x_0)}(h) = \Big[ax_0 + b\Big] + \Big[a\Big]h$$

1. The *local constant coefficient*

$$\Big[ax_0 + b\Big]$$

gives the Height-sign near x_0 because it contributes most of the *local height* since, as long as h remains *small*, that is as long as x remains near x_0, the other ters of the local input-ouput rule, namely the *linear term*, is *small* and therefore contributes very little height to the *total local height* and certainly not enough to *change the sign* of the total local height.
And since the local *constant* coefficient *near* x_0 is in fact the output of the function itself *for* x_0, we have:

THEOREM 5 (**Height-sign Near** x_0). *For any affine function* $AFFINE_{a,b}$*:*

- $When\ AFFINE(x_0) = +,$ Height-sign $AFFINE\ |_{\text{near } x_0} = (+,+)$
- $When\ AFFINE(x_0) = -,$ Height-sign $AFFINE\ |_{\text{near } x_0} = (-,-)$
- $When\ AFFINE(x_0) = 0,$ Height-sign $AFFINE|_{near\ x_0}$ *is given by* Sign a

2. The *local linear term*

$$\Big[a\Big]h$$

contributes *all* of the local slope since the constant term has no slope.
The *local* linear *coefficient* is therefore what controls the *local slope* near any finite input x_0:

THEOREM 6 (**Slope-sign Near** x_0). *For any quadratic function* $AFFINE_{a,b}$*:*

- *When Sign* $a = +,$ Slope-sign $AFFINE\ |_{near\ x_0} = (\diagup, \diagup)$
- *When Sign* $a = -,$ Slope-sign $AFFINE\ |_{near\ x_0} = (\diagdown, \diagdown)$

3. Using the *standard notation* in which we think of

$\diagup$ as being *positive*, that is as $+$
$\diagdown$ as being *negative*, that is as $-$

a more compact way of stating the above theorem is

THEOREM 6 (**Slope-sign Near** x_0). *Given the function* $AFFINE_{a,b}$

- *When Sign* $a = +,$ Slope-sign $AFFINE|_{near\ x_0} = (+,+)$
- *When Sign* $a = -,$ Slope-sign $AFFINE|_{near\ x_0} = (-,-)$