A DEFINITIONS

We borrow and adapt existing definitions of Pareto optimality from Marler & Arora (2004).

Strong Pareto Optimal: A point $\tilde{x}^*$ is *strong* Pareto optimal if there does not exist another $x_j \in X$, *s.t.* $f_p(x_j) \leq f_p(x^*)$ for all functions f_p and $f_l(x_j) < f_l(x^*)$ for at least one function f_l:

$$\nexists x_j : f_p(x_j) \leq f_p(x^*), \quad \text{for} \quad p = 1, 2, \ldots, k \quad \exists l : f_l(x_j) < f_l(x^*) \tag{6}$$

In other words, no point exists that improves at least one objective without detriment to other objectives.

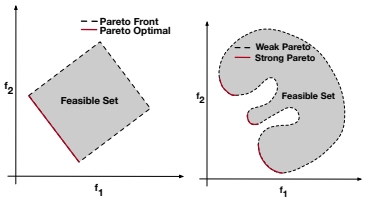
Weak Pareto Optimal: A point $\tilde{x}^*$ is *weak* Pareto optimal if no other point exists that improves all of the objectives simultaneously.

$$\nexists x_j : f_p(x_j) < f_p(\tilde{x}^*), \quad \text{for} \quad p = 1, 2, \ldots, k \tag{7}$$

This is different from strong Pareto, where points might exist which improves at least one objective at no detriment to others.

Dominated and Non-Dominated Points: Dominated points are defined *w.r.t.* the objective functions. The objective function vector $F(x^*)$ is non-dominated (*strong*) *iff* no other vector $F(x)$ exists *s.t.* $F(x) \leq F(x^*)$ with at least one $f_p(x) < f_p(x^*)$. The vector $F(x^*)$ is considered dominated (*weak*) otherwise.

Fig. 7 shows two MOOs for two competing functions. The feasible set of points are shown as gray shaded area with the Pareto boundary shown as dashed lines. For both problems, a joint minimization problem is considered, resulting in a Pareto optimal set (red curves) facing the origin. A different Pareto boundary can be obtained if a joint-maximization or a mixed min-max problem is considered. For Fig. 7 (a), all marked points form the *strong* or non-dominated set. For Fig. 7 (b), a strong Pareto optimal (non-dominated) solution set from a non-convex *weak* or dominated Pareto front can result in a discontinuous manifold.



(a) Continuous (b) Discontinuous

Figure 7: Pareto optimal set under different objectives. Note that the red line corresponds to the min-min strong Pareto optimal front for both problems and the dashed line corresponds to the *weak* Pareto Front for the entire feasible set.

B THE NECESSITY OF FRITZ-JOHN CONDITIONS (FJC) AND MISSTATED INFERENCES ON LINEAR SCALARIZATION

Consider a simple problem with two psuedo-convex, Gaussian functions in a variable domain $x \in \mathbb{R}^n$:

$$\begin{aligned} f_1(x_1, \ldots, x_n) &= 1 - exp(-[(x_1 - 1/\sqrt{(n)})^2 + \ldots + (x_n - 1/\sqrt{(n)})^2]) \\ f_2(x_1, \ldots, x_n) &= 1 - exp(-[(x_1 + 1/\sqrt{(n)})^2 + \ldots + (x_n + 1/\sqrt{(n)})^2]) \\ \text{s.t.} \quad & g_1, \ldots, g_n : -1/\sqrt{n} \leq g_i \leq 1/\sqrt{n}, \ \forall i \in [1, n] \end{aligned}$$

For ease of visualization we consider $n = 1$ where the inferences can be extended to higher dimensional variable domain by increasing n. The MOO problem then reads:

$$\min_x \quad \vec{F}(x) = (f_1(x), f_2(x)) \quad \text{s.t.} \quad -1 \leq x \leq 1, x \in \mathbb{R}$$

However, the MOO problem is not directly solvable since an optimization approach requires a scalar objective function as opposed to the vector objective $\vec{F}(x)$. A linear scalarized function composed of the two original objectives $f_1(x)$ and $f_2(x)$ weighted by α can then be written as:

$$S(\alpha, x) = \alpha f_1(x) + (1 - \alpha) f_2(x).$$

Before we delve into the Pareto manifold, let us take a moment to discuss the surface geometry of this scalarized function $S(\alpha, X)$. **Fig. 8 (a)** shows the surface geometry of $S(\alpha, x)$ where one can easily see that the function $S(\alpha, x)$ has *three stationary points w.r.t.* to an optimization problem (gradient

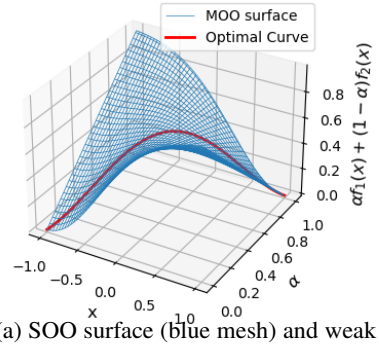
vector is identically zero): **(1)** minimum at $x = -1$ and $\alpha = 0$, **(2)** minimum at $x = 1$ and $\alpha = 1$, and **(3)** a saddle point at $x = 0$ and $\alpha = 0.5$.

Now if we solve, $\underset{\alpha,x}{min} S(\alpha, x)$ we can find either of the two minima of $S(\alpha, x)$ noted above using different initializations. On the other hand if we solve: $\underset{x}{min}\,\underset{\alpha}{max}\, S(\alpha, x)$ we will arrive at the aforementioned saddle point of $S(\alpha, x)$. However, the Pareto manifold is the entire red curve in shown in **Fig. 8 (a)** and solving the above two problems can lead us to only three points in the Pareto optimal solution set. **Fig. 8 (b)** shows the Pareto optimal solution set (blue curve) in a functional domain with the red dots on the blue curve indicating the three stationary points of $S(\alpha, x)$. One can easily verify that the gradient vector $\nabla_{\alpha,x} S(\alpha, x)$ is not identically zero at any other point on the red curve in **Fig. 8 (a)** apart from the three aforementioned stationary points.

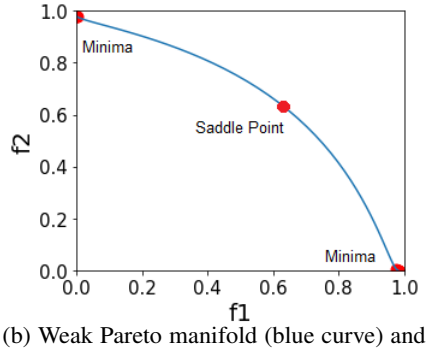
Remark. *Any optimization solver that relies upon gradient of $S(\alpha, x)$ can only converge to the stationary points of $S(\alpha, x)$ where the gradient becomes identically zero or the set of points $\{(\alpha, x) | \nabla_{\alpha,x} S(\alpha, x) = \mathbf{0}\}$. Therefore expecting that a modified optimization over linear scalarized function $S(\alpha, x)$ will retrieve the entire Pareto set is not correct.*

What if we partitioned the domain (cone, bins, rays *etc.*) into sub-domains and searched locally? Consider the bin $\alpha \in [0.1, 0.4]\, \forall x$, with the Pareto optimal solution manifold restricted to that block in **Figs. 8 (a)** (red curve) and **(b)** (blue curve). The issue still remains unchanged: for any optimization problem on $S(\alpha, x)$ restricted to the above bin (or chunk) there is no point (α, x) where the gradient vector is identically zero (stationary point). Therefore no optimization approach with $S(\alpha, x)$ objective can converge to the Pareto manifold in this region. Note that for an optimization solver to converge to a user-prescribed tolerance the norm of the gradient vector must fall below this tolerance. *So while MTL methods (Sener & Koltun, 2018; Lin et al., 2019b; Mahapatra & Rajan, 2020; Navon et al., 2021) rely upon min-norm solver, the Pareto optimal solution set consists of points where any norm over gradient vector at these points will not be zero.*

So how can we converge at the Pareto optimal solution points that are not the stationary points of $S(\alpha, x)$? The answer is Fritz-John conditions (FJC) written in the determinant form that results in a different scalar function $D(x) = det(L^T L)$ where the set of points $\mathcal{P} = \{x | D(x) = 0\}$ overlaps the Pareto optimal set. As mentioned before, the $\mathcal{P}$ might still contain dominated points and therefore a filter might be necessary, subjective to the problem at hand, to obtain the Pareto optimal set.



(a) SOO surface (blue mesh) and weak Pareto manifold (red curve).



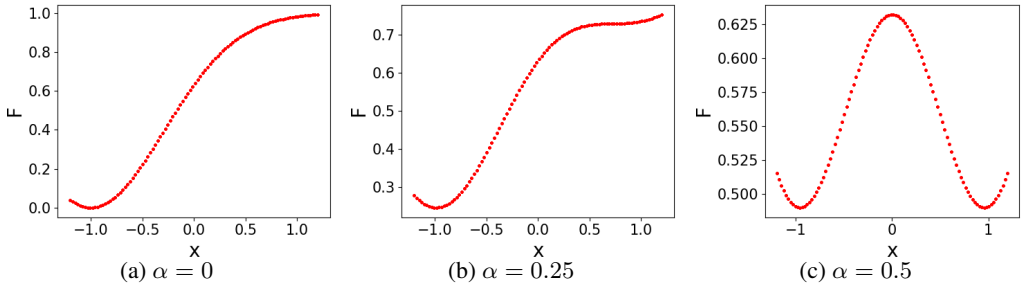
(b) Weak Pareto manifold (blue curve) and stationary points of the SOO problem.

Figure 8: Pareto surface for the Gaussian benchmark case. (a) MOO saddle point trajectory of the problem showing strong attractors at $x = \pm 1$ for $\alpha = 0, 1$ respectively. (b) Pareto points produced by Linear Scalarization collapses to the individual minima for f_1, f_2 for 10 different runs. While the extreme red points are the solution of the $\underset{x,\alpha}{min}\, S(\alpha, x)$, the red point in the middle is the saddle point solution of $\underset{\alpha}{max}\,\underset{x}{min}\, S(\alpha, x)$.

Remark. *The claim made in MTL works Lin et al. (2019a) (Section 5: Synthetic Example), Mahapatra & Rajan (2020) (Section 3.2: Limitations of current methods), Navon et al. (2021) (Section 2: Multi-objective optimization) is: if the Pareto front is non-convex, LS would always fail, wherein a reference to Boyd et al. (2004) is given for proof. However, it is not clear as to which proof in Boyd et al. (2004) is being referred to.*

Although the two functions are psuedo-convex, the Pareto front in the functional domain is non-convex. Here, optimization of an LS objective fails to find the complete Pareto set as shown in **Fig. 8**

(a) (red curve). Contrary to the claims made in MTL works that the non-convex nature of the front is responsible for the failure of LS, we show the stationary points (minima, maxima, saddle points) of an LS objective are only a subset of the Pareto optimal set. Note that a minimization problem on the linearly scalarized function: $S : \alpha f_1(x) + (1-\alpha) f_2(x)$ in **Fig. 8 (a)** can only return the two minima located at $(x = -1, \alpha = 0)$ and $(x = 1, \alpha = 1)$. Therefore, it is incorrect to state that LS itself fails if the Pareto front is non-convex as mentioned in Lin et al. (2019a); Mahapatra & Rajan (2020); Navon et al. (2021). LS is guaranteed to produce an even spread of points *only when* the functions and constraints are convex in the variable domain.



(a) $\alpha = 0$ (b) $\alpha = 0.25$ (c) $\alpha = 0.5$

Figure 9: Slices of $S(\alpha, x)$ surface along α. Depending on α, the linear scalarized SOO problem always has two minima either at $x = -1, 1$ or both. LS works as intended in this scenario converging to these minima. The non-convexity of the front is not responsible for LS to stick at $\approx \pm 1$ in Fig. 8.

Fig. 9 presents a different perspective of this linear scalarized SOO problem. Instead of optimizing $S(\alpha, x)$ over (α, x), consider slices of $S(\alpha_i, x$ where i is the slice index along the α dimension. If we now gather the optima (maxima/minima) along each α slice we can extract the entire Pareto solution set using LS where the number of Pareto optimal points are equal to the number of slices. **Why do we have stationary points at the Pareto front now?** Restricted to an α value, $S(\alpha = 0.5, x)$ becomes a function of x only and now the gradient of the function $S(0.5, x)$ at $x = 0$ is zero. In other words, $x = 0$ is a stationary point (local maximum) of the function $S(\alpha = 0.5, x)$.

C Analysis of HNPF *vs.* OR and MTL methods

MTL methods rely on LS and domain decomposition to compute the Pareto set, with the utopia/ideal point to serve as reference point for their respective solver to work. A cone-dividing or ray-tracing approach will have increasing difficulty in extracting the entire front as: **i)** the number of functions increase; and **ii)** the number of extracted Pareto points increases. HNPF explicitly uses a FJC guided discriminator to find the Pareto set, without relying upon an ideal/utopia point. The FJC discriminator identifies all points where at least one gradient of the multiple objectives is zero as a potential Pareto optimal candidate. HNPF therefore finds all four fronts (max-max, min-max, min-min and max-min), given they exist within the feasible domain $\mathcal{H}$. Existing OR and MTL methods can only solve for either one these four sub-problems (max-max, min-max, min-min and max-min) since they need the domain decomposition and solution trajectory *w.r.t.* the reference point for each of the four sub-problems.

To illustrate this claim, we consider two functions f_1 and f_2 and three constraints g_1, g_2, g_3, that jointly define a Pareto front. Note that although the functions f_1, f_2 themselves are non-competing, the non-convex constraint g_1 defines the Pareto set, hence the unique circular front. The feasible set $\mathcal{H}$ is defined by constraints g_2, g_3 serving as bounding box. Jointly optimize:

$$f_1(x_1, x_2) = x_1, \quad f_2(x_1, x_2) = x_2$$
$$\text{s.t.} \quad g_1(x_1, x_2) : (x_1 - 0.0)^2 + (x_2 - 0.0)^2 \leq 1.0$$
$$g_2, g_3 : -1 \leq x_1, x_2 \leq 1$$

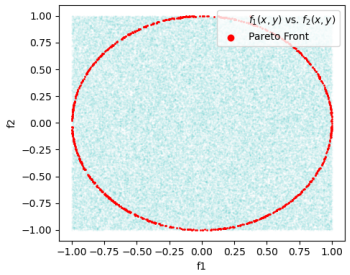


Figure 10: HNPF's weak Pareto set.

Fig. 10 shows the HNPF extracted *weak* Pareto set where the first, second, third and fourth quadrant represents the max-max, min-max, min-min and max-min Pareto sets. Note that without any reliance on the reference point, HNPF has the capacity to optimize the MOO problem and extract the entire

weak Pareto front. Now depending on the user requirement, one can apply the Pareto filter to extract the set corresponding to any one of the sub-problems. Any OR or MTL (if the MTL method can support constraints) methods can only recover one quadrant of the entire front due to their design and dependence on the reference point.

D DIFFERENCE BETWEEN HNPF *vs.* PHN

HNPF (Stage-1) extracts the *weak* Pareto manifold as an n-dimensional diffusive indicator function as opposed to a $(n-1)$-dimensional manifold itself, where the regressed manifold is not only guided by the weak Pareto points (indicator value 1) but also the sub-optimal points (indicator value 0) for a more robust and accurate extraction. Please refer to illustrations in **Section 5** and **Appendix I**.

In comparison, PHN (Navon et al., 2021) uses a neural network to regress over the $(n-1)$-dimensional manifold in the variable domain directly, given solution points obtained from EPO or LS. However:

i) Neither EPO nor LS are guaranteed to work for non-convex scenarios. Readers are referred to **Section 5** and **Table 2** for correctness and applicability of EPO on different benchmark cases.

ii) If the non-dominated Pareto set is discontinuous, then PHN will wrongly identify non-Pareto points as Pareto optimal. Consider **Fig. 11** where neither LS or EPO works. Even if we provided the true discontinuous *strong* Pareto set for PHN to regress over the resulting $n-1$ dimensional manifold will be smooth wrongly indicating that Pareto points exist in the regions marked by red circles. Since HNPF learns a manifold with support on an n-dimensional variable domain as a diffusive indicator by using both the Pareto and non-Pareto point, followed by a Pareto Filter, this situation never arises (see **Fig. 5.4**).

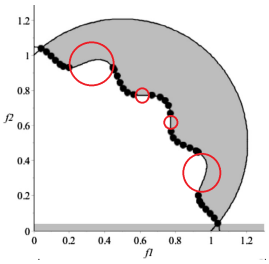


Figure 11: PHN Results on a Hypothetical correct disconnected *strong* Pareto set.

iii) Even when the Pareto optimal set is a continuous manifold, PHN inherently assumes that the point spread obtained from EPO or LS is uniformly distributed over the true manifold for the neural regressor to not overfit. Please refer to Case II (**Fig. 5.3**) where, for all the regions EPO fails (converges to sub-optimal points), PHN learns the wrong manifold and generates sub-optimal points.

iv) If the manifold is an implicit surface in the functional domain, a direct regression using a neural network is not feasible. See **Fig. 3** where the weak Pareto front is self intersecting implicit surface.

E ISSUE WITH PARETO FRAMING OF MTL NETWORKS

Sener & Koltun (2018) used LeNet for Multi-MNIST classification. All successive MTL methods use this setup for demonstrating their ability to find the Pareto points of this MOO problem. However, there are two issues with this setup from a Pareto framing point of view that have not been justified.

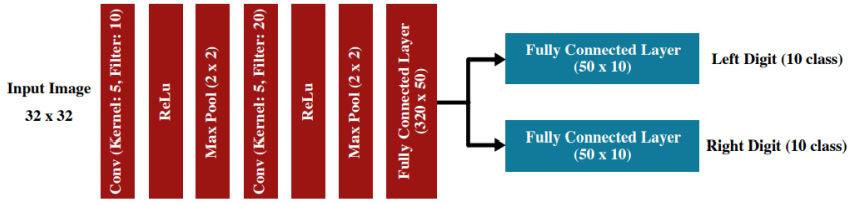


Figure 12: Modified LeNet architecture used in MTL works.

i) Finding Pareto points between spaces with partially shared variable domain: Consider the scenario in **Fig. 13**, which is applicable for **Fig. 12** or any MTL framework presented in Sener & Koltun (2018). Let $\theta_{sh} \in \mathbb{R}^c$ be the weights of the shared layer, and $\theta_{t1} \in \mathbb{R}^d$, $\theta_{t2} \in \mathbb{R}^d$ be the weights of the task specific layers respectively. Finding Pareto optimality over the shared weights θ_{sh} results in two competing objectives to trade-off. However, Pareto optimality over $(\theta_{sh}, \theta_{t1})$ and $(\theta_{sh}, \theta_{t2})$ where θ_{t1} and θ_{t2} are independent of each other is still equivalent to competing over the shared variable domain θ_{sh} only. It is well know that introducing additional weights (θ_{t1} and θ_{t2}) increases the expressive power of a neural network (Csáji et al., 2001). However, this modified network cannot be compared with the original shared only network since each of the task-specific

(independent) weights can now be learned without detrimentally affecting the other (no-competition or trade-off). Hence the network can achieve better values of objectives (notice the MTL objective values exceeding that of the individual SOO values in **Fig. 14**).

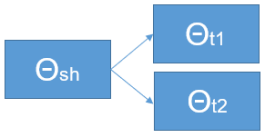
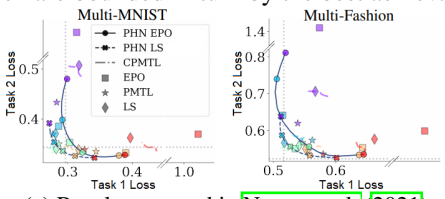
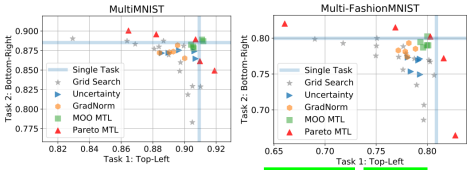


Figure 13: MTL Framework containing a shared weights θ_{sh} and task-specific weights θ_{t1} and θ_{t2}.

ii) MOO value exceeding individual SOO value: We elaborate on the incompleteness and often wrong inferences, violating Pareto definitions on practical datasets for MTL methods. **Fig. 14** shows obtained results (figures borrowed from Navon et al. (2021) and Lin et al. (2019a)) from MTL methods on Multi-Mnist (Deng, 2012) and Multi-FashionMnist (Xiao et al., 2017) datasets. Given two functions f_1, f_2 and user trade-off α, the scalarized SOO problem is: $S = \alpha f_1 + (1-\alpha) f_2$. LS would simply try to solve this optimization problem for different prescribed α's using gradient descent and converge to some global/local optima. Nonetheless, it is still solving F, the extremum of which are bounded in turn by the best achievable f_1 and f_2 respectively.



(a) Results reported in Navon et al. (2021)



(b) Results reported in Lin et al. (2019a)

Figure 14: Inference of MTL methods on practical datasets. For all four graphs, even in a competing setting, MTL methods are able to find solution points that exceed the best possible single objective results obtained.

Remark. *Under a competing setting, all solutions of the SOO problem $S : \alpha f_1 + (1-\alpha) f_2$ for values of $\alpha \in (0, 1)$ is bounded by $f_1(\alpha = 1)$ and $f_2(\alpha = 0)$ respectively. If any solution exceeds f_1 or f_2, the only implication is that the functions are not competing, rather helping each other in the joint setting, which is not the Pareto problem.*

For **Fig. 14 (a)**, the vertical and horizontal dashed lines represent the best obtained single objective value (*i.e.* solving for f_1 and f_2 independently, corresponding to the MOO setting for $\alpha = 0/1$). Similarly, for **Fig. 14 (b)** the vertical and horizontal turquoise lines represent the best obtained single objective value. While Fig. 14 (a) considers minimizing the classification loss value, Fig. 14 (b) considers maximizing the classification accuracy on the Multi-Mnist and Multi-FashionMnist datasets respectively. The reader is directed to the Pareto solution points for MTL approaches, where in a competing setting due to the alteration of the LeNet network itself, their MOO solution achieves better functional value than the respective single objective optima.

Furthermore, all the MTL solvers are claimed to be correct. An inference of correctness of two Pareto solver is that the front produced by both would be the same irrespective of the density of Pareto points produced by each. **Fig. 14** violates the proposed correctness claims by producing multiple fronts for a given dataset and objectives. A practitioner's tendency is to solve their practical problem, if tools do exists in literature. As such it is the duty of the tool developer to necessitate caution if such tool was developed under a convex setting, but being applied to a practical non-convex problem. Deviation from convexity will result in spurious results and any inferences made on them will be erroneous. When these problems are of significant societal consequence, extreme caution is warranted.

F Error Bound

For a user-specified relaxation margin $0 \leq \epsilon \leq 1$, the approximation error between the network extracted manifold $\tilde{M}(\tilde{X})$ and the true solution $M(X^*)$ is bounded below by $\|\tilde{M}(\tilde{X}) - M(X^*)\|_2 \leq \epsilon$. Assuming the L matrix from Eq. 4 is square, we have $det(L) = 0$ (see **Appendix G** for equivalence). From Leibnitz formula for determinants:

$$det(L) = det\Big(\begin{bmatrix} \nabla F & \nabla G \\ \mathbf{0} & G \end{bmatrix} \Big) = det(\nabla F) det(G) = 0$$

Further assume that,

$$|det(L(\tilde{x}))| \leq \epsilon,\ \tilde{x} \neq x^* \tag{8}$$

where $\epsilon > 0$, and x^* and $\tilde{x}$ are the optimal points and the network generated approximate solution points, respectively. The Fritz John necessary conditions in Eq. 2 for *weak* Pareto optimality is:

$$det(L(x^*)) = 0 \tag{9}$$

Combining the assumption in Eq. 8 and Eq. 9, we have

$$|det(L(\tilde{x})) - det(L(x^*))| \leq \epsilon \tag{10}$$

The *weak* Pareto optimal solution manifold is given by $M(X) = det(L(X)) = 0$. Assume an approximate manifold $M(\tilde{X})$ such that:

$$\|M(\tilde{X}) - M(X^*)\|_2 \leq \epsilon \tag{11}$$

When the network converges, Eq. 11 will hold for the network approximated $\tilde{M}(\tilde{X})$. Here, $\tilde{x} \in \tilde{X} = \{x|\tilde{M}(\tilde{X}) \leq \epsilon\}$ and X^* is the set of true optimal points such that $M(x^*) = 0, \forall x \in X^*$. Since we explicitly specify ϵ in our loss, we know that the network generated solution is ϵ close to $M(\tilde{X})$ if:

$$\|M(\tilde{X}) - \tilde{M}(\tilde{X})\|_2 \leq C\epsilon, \quad 0 \leq C \leq 1 \tag{12}$$

Eq. 12 implies that if we are able to find such a C, then we implicitly satisfy Eq. 11. Hence,

$$\|\tilde{M}(\tilde{X}) - M(X^*)\|_2 \leq \epsilon \tag{13}$$

G EQUIVALENCE OF $det(L) = 0$ AND $det(L^T L) = 0$

The $det(L)$ matrix defined in Eq. 4 is given by:

$$L = \begin{bmatrix} \nabla F & \nabla G \\ \mathbf{0} & G \end{bmatrix}$$

To achieve $det(L) = 0$ requires that either:

1. $\nabla F(x) = 0$: atleast one objective function has reached its optimum (local/global minima/maxima under a min/max setting); *and / or*
2. $G(x) = 0$: at least one constraint is satisfied.

This criteria is only applicable for square systems. However, for practical problems, the system might become non-square, hence we need to satisfy $det(L^T L) = 0$ following Eq. 5. One might think that it's a different optimization problem. However mathematically satisfying $det(L) = 0$ is equivalent to satisfying $det(L^T L) = 0$ and we provide the derivation of it.

$$\begin{aligned} det(L^T L) &= \begin{bmatrix} \nabla F^T & \mathbf{0} \\ \nabla G^T & G^T \end{bmatrix} \begin{bmatrix} \nabla F & \nabla G \\ \mathbf{0} & G \end{bmatrix} \\ &= \begin{bmatrix} \nabla F^T \nabla F & \nabla F^T \nabla G \\ \nabla G^T \nabla F & \nabla G^T \nabla G + G^T G \end{bmatrix} \end{aligned} \tag{14}$$

We now observe Eq. 14 for the two cases prescribed above and see if $det(L^T L)$ evaluates to zero or not. For Case 1, where $\nabla F = 0$, Eq. 14 reduces to:

$$det(L^T L) = \begin{bmatrix} \mathbf{0} & \mathbf{0}\nabla G \\ \nabla G^T \mathbf{0} & \nabla G^T \nabla G + G^T G \end{bmatrix}$$

which is low-rank since row 1 equates to 0. For Case 2, where $G = 0$, Eq. 14 reduces to:

$$\begin{aligned} det(L^T L) &= \begin{bmatrix} \nabla F^T \nabla F & \nabla F^T \nabla G \\ \nabla G^T \nabla F & \nabla G^T \nabla G + 0 \end{bmatrix} \\ &= \nabla F^T \nabla G^T \begin{bmatrix} \nabla F & \nabla G \\ \nabla F & \nabla G \end{bmatrix} \end{aligned}$$

which is low-rank again because row 1 and row 2 are equal. Hence proven that satisfying $det(L) = 0$ is equivalent to satisfying $det(L^T L) = 0$.

H MODELER INTERPRETABILITY

Motivated by Lipton (2018)'s definitions of model interpretability and trust, we adopt the persona of a modeler in assessing the interpretability of our model. In all of the problems above, the approximate manifold $\tilde{M}$ is described by the user specified loss function. If a domain specific analytical solution ($M(X^*) = 0$) is known, then the approximate network generated solution set ($\tilde{M}(\tilde{X}) \leq \epsilon$) can be verified by comparing $\tilde{X}$ and X^*. Additionally, a domain-specific modeler can also compare the approximate manifold $\tilde{M}$ (**Fig. 15** for **Case III**), at the last layer of the network, against the true manifold M known from the analytical form. If the modeler is able to verify that the network classifies the correct (truly Pareto optimal) data points in the variable space as being Pareto optimal (high probability value), the trust in the network's working is established.

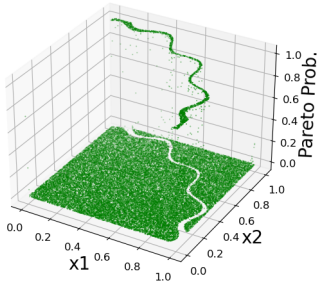
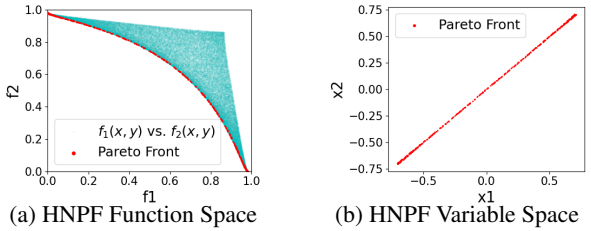


Figure 15: Classification boundary of *weak* Pareto points for Case III. The final layer assigns a probability score to each point in the variable space as being Pareto optimal or not. A modeler needs to examine the produced points/manifold against the known analytical form in the variable domain to verify the correctness of the network's working.

I ADDITIONAL BENCHMARKS

I.1 CASE I: ($n = 2, k = 2, m = 2$)

Fig. 16 shows the HNPF extracted Pareto front in both functional and variable domain.



(a) HNPF Function Space (b) HNPF Variable Space

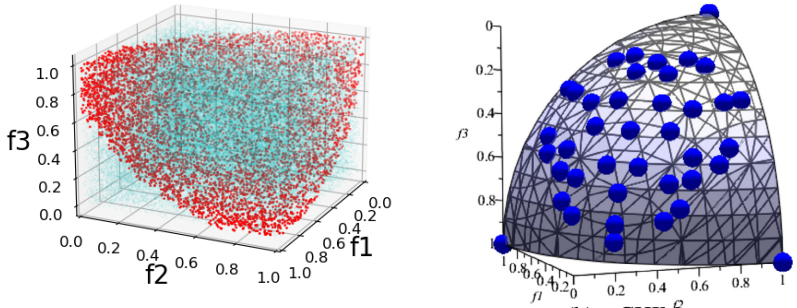
Figure 16: Pareto Front for Case I. Since the functions and constraints are convex, all OR and MTL methods work here, producing Pareto points with varying density.

I.2 CASE V: CONVEX OBJECTIVES, NON-CONVEX CONSTRAINT ($n = 3, k = 3, m = 4$)

This problem was proposed in Ghane-Kanafi & Khorram (2015). Jointly minimize

$$f_1(x_1, x_2, x_3) = x_1, \quad f_2(x_1, x_2, x_3) = x_2, \quad f_3(x_1, x_2, x_3) = x_3$$
$$\text{s.t.} \quad g_1(x_1, x_2, x_3) : (x_1 - 1)^2 + (x_2 - 1)^2 + (x_3 - 1)^2 \leq 1.0$$
$$g_2, g_3, g_4 : x_1, x_2, x_3 \geq 0$$

This form is convex in f_1, f_2, f_3 but the non-convex constraint in g_1 forces the Pareto front to be non-convex. The result using our HNPF method, as shown in Fig. 17, is in good agreement with mCHIM method but with a significantly higher point density.



(a) HNPF Function Space (b) mCHIM

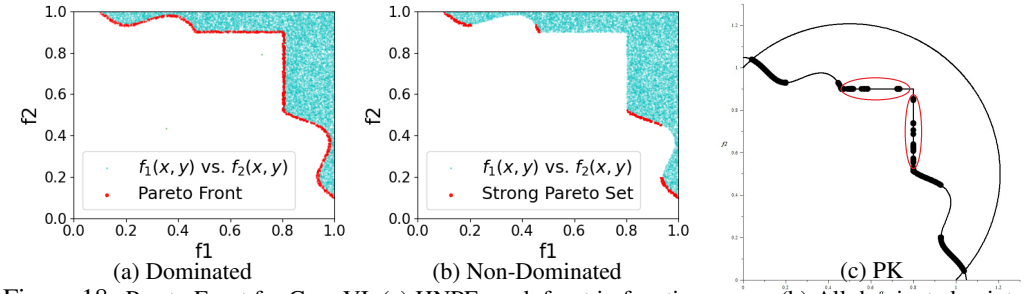
Figure 17: Pareto Front for Case V. The axis orientation in anti-clockwise order are a) HNPF f_3, f_2, f_1, and (b) mCHIM f_3, f_1, f_2 respectively.

I.3 CASE VI: N=2, K=2, M=5

This form is an extension of Case III, with an additional *max* boundary constraint g_3. PK computes the true Pareto front with limited density $n = 40$ points. Fig. 18 (a) shows the *weak* Pareto front with dominated points. As before, after post-processing with the proposed Pareto filter we arrive at the Pareto set with non-dominated points shown in Fig. 18 (b).

This problem was proposed in Dutta & Kaya (2011). Jointly minimize

$$
\begin{aligned}
& f_1(x_1, x_2) = x_1 \\
& f_2(x_1, x_2) = x_2 \\
& \text{s.t.} \quad g_1(x_1, x_2) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5 \\
& g_2(x_1, x_2) = x_1^2 + x_2^2 - 1 - 0.1\cos(16\arctan(\frac{x_1}{x_2})) \geq 0 \\
& g_3(x_1, x_2) = max(|x_1 - 0.6|, |x_2 - 0.7|) - 0.2 \geq 0 \\
& g_4, g_5 : 0 \leq x_1, x_2 \leq \pi
\end{aligned}
$$



(a) Dominated (b) Non-Dominated (c) PK

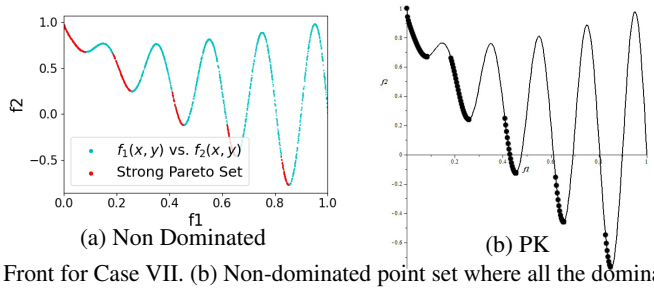
Figure 18: Pareto Front for Case VI. (a) HNPF weak front in function space. (b) All dominated points are removed from the set after application of Pareto filter. PK generates points with low density. Note that PK is not able to remove some of the dominated points on the weak front (red circled).

I.4 CASE VII: N=30, K=2, M=30

This problem was proposed in Pirouz & Khorram (2016), albeit with a discrepancy[3]. Jointly minimize

$$
\begin{aligned}
& f_1(x) = x_1 \\
& f_2(x) = f(x)\left(1 - \left(\frac{f_1(x)}{f(x)}\right)^{0.5} - \frac{f_1(x)}{f(x)} sin(10\pi x_1)\right) \\
& \text{s.t.} \quad f(x) = 1 + \frac{9}{m-1}\sum_{i=2}^{n} x_i^2 \\
& g_1, \ldots, g_{30} : x_1 \in [0, 1], x_i \in [-1, 1], \forall i = 2, \ldots, m
\end{aligned}
$$

[3]Although the normalization term proposed in $f(x)$ is $m - 1$, it does not generate the curve reported in Pirouz & Khorram (2016). We were able to replicate the shown curve, in our experiments, by choosing a normalizing constant of 10000.



(a) Non Dominated (b) PK

Figure 19: Pareto Front for Case VII. (b) Non-dominated point set where all the dominated points are removed from the set after application of Pareto filter. Also note the low density of points in PK.

This form is convex in f_1 and non-convex in f_2. The dimension of the design variable space is $m = 30$. The corresponding Pareto front is non-convex. The results using our method, as shown in **Fig. 19**, are in good agreement with PK methos. Even in this high-dimensional setting, we obtain a *weak* Pareto front with high point density as shown in Fig. 19 (a). The distribution of the objective space in this setting is such that the entire space is the front itself. Hence, we cannot see the cyan points in Fig. 19 (a). As before, after post-processing with the proposed Pareto filter we arrive at the Pareto set with non-dominated points shown in Fig. 19 (b).

J Implementation Details

J.1 Setup

Experiments use a Nvidia 2060 RTX Super 8GB GPU, Intel Core i7-9700F 3.0GHz 8-core CPU and 16GB DDR4 memory. We use the Keras (Chollet, 2015) library on a Tensorflow 2.0 backend with Python 3.7 to train the networks in this paper. For optimization, we use AdaMax (Kingma & Ba, 2014) with parameters (*lr*=0.001) and 1000 steps per epoch. Stage-1 of HNPF is a feed-forward dense neural network: three layers of dense connections with eight neurons each and *tanh* activation.

J.2 Error Tolerance

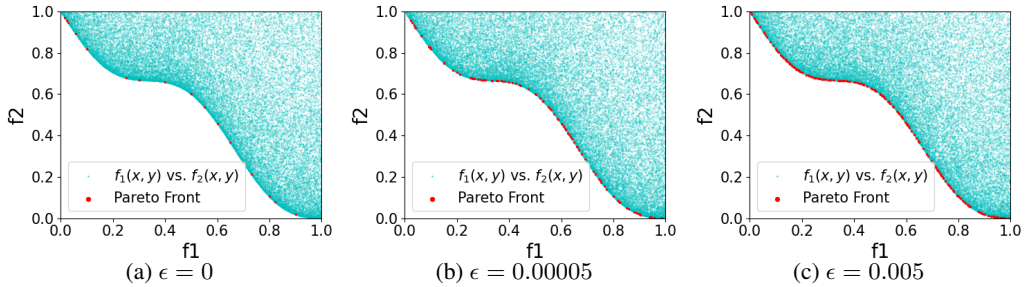
A strength of HNPF is that all network weights can be initialized randomly, yet it would arrive at the same solution every run. Since the data domain is discrete, an exact zero might not be achievable. We therefore use a slightly relaxed criterion of $\epsilon = 0.0005$ as the classification margin. Any point below this value will be classified as *weak* Pareto. For all results, the extracted Pareto set $\tilde{M}(\tilde{X})$ (shaded red) overlaps the true Pareto set $M(X^*)$ with an ϵ spread.

Due to stochastic variation, neural network studies often report variance across several runs. However, the only approximation errors with HNPF lie in the extracted manifold over runs. Since the true manifold $M(X^*)$ remains constant across runs, the loss itself is the approximation error of the deviation of the extracted manifold $\tilde{M}(\tilde{X})$ with a minimum achievable value of 0 at machine precision. We therefore do not report mean-variance across runs. Additional ablation studies on nature of the extracted front *w.r.t.* network size, choice of ϵ and training density are elaborated in **Appendix K**. Please refer to **Appendix L** for the training and validation loss profiles.

K Ablation Study

K.1 Effect of Relaxation Parameter ϵ

As mentioned before, since the data domain is discrete, as $\epsilon \to 0$ in Eq. 5, the number of solution points decreases, as expected. **Fig. 20** shows the variation in the density of front *w.r.t.* the user chosen ϵ. Higher the value of ϵ, more relaxed is the *weak* Pareto classification margin, hence the density of points within that margin increases linearly with increasing ϵ. This effect corroborates our presented error bounds between the true Pareto manifold $M(X^*)$ ($0 \leq x_1 \leq 1, x_2 = 0$ in this case) *vs.* the neural network approximated manifold $\tilde{M}(\tilde{X})$.

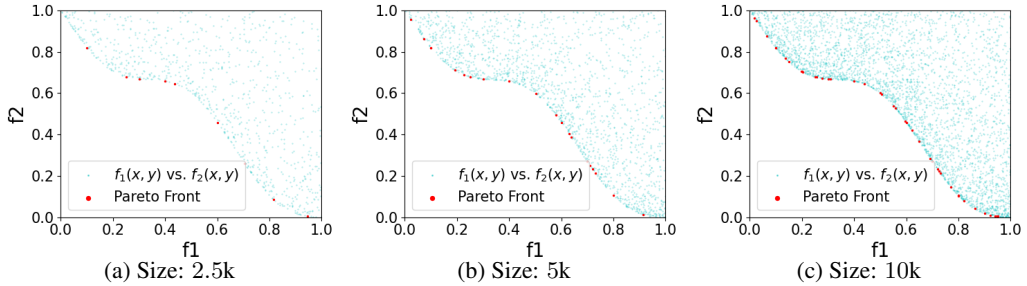


(a) $\epsilon = 0$ (b) $\epsilon = 0.00005$ (c) $\epsilon = 0.005$

Figure 20: Pareto Front for Case II under different choice of ϵ. As the value of ϵ increases, we get an increase in the point density along the true Pareto manifold within an ϵ margin.

K.2 Effect of Training Data Density

Since the input data is discrete, a sufficient point density is necessary to find a good approximation $\tilde{M}(\tilde{X})$ to the true Pareto manifold $M(X^*)$. In **Fig. 21** we illustrate this variation. As long as HNPF approximates a form $\tilde{M}(\tilde{X})$, which is close to the true manifold $M(X^*) : 0 \leq x_1 \leq 1, x_2 = 0$, the generated front will always be within ϵ margin.

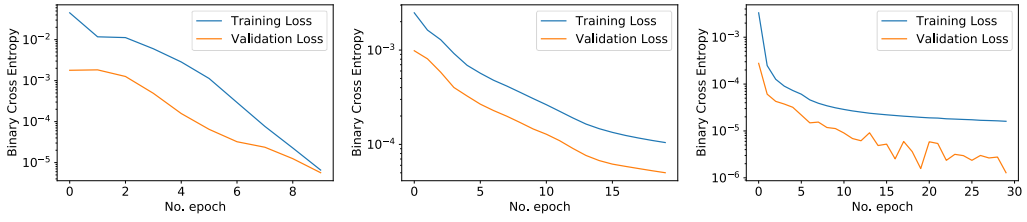


(a) Size: 2.5k (b) Size: 5k (c) Size: 10k

Figure 21: Pareto Front for Case II for different choices of training data density. The manifold approximation improves as the point density is increased in (a) thru (c).

L Loss Profile

We now briefly discuss the training process for the cases shown above. On an average, the network takes around $10/20/30$ epochs for the simple/moderate/hard cases as visualized in **Fig. 22**. Since the last layer of the network is classifying points as being *weak* Pareto or not, the runtime is dictated by the complexity of the curve in the design variable space. The more non-linear the solution manifold, the more training time is required to approximate it.



(a) Case II/IV (b) Case III (c) Case IV

Figure 22: Training (blue) and validation (orange) curves for 4 cases. An increase in the dimensions of the design variable space results in increased costs for constructing the L matrix. Consequently, the network takes more epochs to converge.

Case II and V both converge within 10 epochs although they lie in 2D and 3D space, respectively. Per Fig. 4 and 17 (b), the design variable space is convex and so the solution manifold is less complicated. Although in 2D variable space, Case III takes 20 epochs owing to the sinusoidal solution manifold. Case IV converges in 30 epochs, the design space is 30 dimensional, hence the compute complexity increases due to the construction of a larger L matrix. The validation loss curve lies below the training loss (but strictly at scale), suggesting that our low-weight network did not over/underfit.

M RUNTIME COMPARISON

Using numerical experiments, we previously verified that mCHIM, PK and HNPF approaches arrive at the correct results for all the considered cases. We now perform a compute time analysis against mCHIM and PK, to demonstrate improved performance using our proposed approach. The trajectories in **Fig. 23** show the compute times for the high dimensional **Case VII**. Note that for mCHIM and PK, the timings are reported for dimensions n=30 and n=4, respectively. For our method, the runtimes are reported for Case V with the variable space dimension ranging from $[2-30]$. Note that all MTL methods crash here and are therefore not reported.

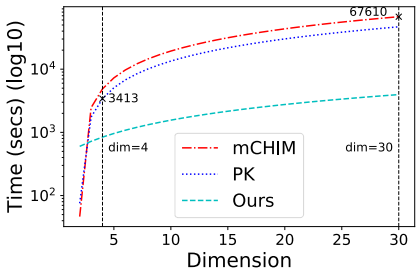
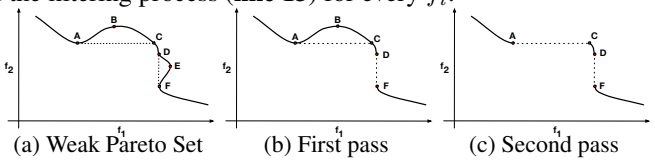


Figure 23: Runtime of HNPF *vs.* mCHIM and PK, as the variable dimension increases. All methods have a linear increase in runtime with dimension, but HNPF scales much better.

The reported runtime with two dimensional variable space might give the false notion that mCHIM and PK are more efficient than HNPF. However, as the variable dimension increases, both mCHIM and PK become far more expensive, as shown in Fig. 23. These methods also produce a low density of Pareto points (~ 40), while HNPF yields high density $(\sim 1k)$. Since both mCHIM and PK are based on enhanced scalarization, solving the resulting problem to extract Pareto points suffers from scaling issues.

N WORKING OF PARETO FILTER

The algorithm starts with the set of all *weak* Pareto points $p \in \mathcal{P}$, which will be refined through the iterative process. The loop (**line 4**) iterates over all the functions f_i. It checks for set of dominating and non-dominating points for all discretization levels (**line 6**) and appends them to a temporary list (**line 10**). If multiple points do exist (**line 11**) for a given level (cardinality > 1), then there certainly are dominated points. The non-dominated point is one which has the lowest function value for the next function $f_q, q = i + 1$. This (**line 12**) states that a point which seems non-dominated for a given function f_i might be dominated for other functions $f_r, r \in k, r \neq i$, but will be taken care of when iterating through function f_r. Once the non-dominated point has been found, it implies that all the other points are in fact dominated, hence should not be considered for further evaluation. They are rejected (**line 13**) from the active set $\mathcal{P} = \mathcal{P} \backslash (temp \backslash x_p)$. The output is the set of strong Pareto points which were all non-dominated for every function f_i, and are essentially the points in the *weak* Pareto set $\mathcal{P}$ that survived the filtering process (**line 13**) for every f_i.



(a) Weak Pareto Set (b) First pass (c) Second pass

Figure 24: Illustration of the proposed Pareto Filter on a two-function min-min setting for visualization.

It is easy to visualize the working of the proposed Pareto filter in **Fig. 24**. We start with the *weak* Pareto set P (Fig. 24 (a)) for a non-convex form. The first pass over f_1 removes a set (segment DEF) of dominated Pareto points (Fig. 24 (b)). The leftover points in P are then filtered again based on f_2 (Fig. 24 (c)), where the dominated points (segment ABC) as per f_2 are removed. Points surviving the filtering process belong to the strong/non-dominated Pareto set.

O DETAILED LITERATURE SURVEY

Since a Pareto solution set reflects optimal trade-off between competing objectives and constraints, the user choice depends on the preferred trade-off value. Prior works can be categorized into four classes of user preferences: 1) *No preference* (Zeleny, 1973): user preference criteria are not explicitly specified; 2) *a priori* (Gal, 1980): preference criteria are explicitly specified before computation; 3) *a posteriori* (Das & Dennis, 1998): preference criteria are explicitly specified after computation; and 4) *Interactive methods* (Miettinen, 2012): preference criteria are continuously consulted to isolate one of the optimal solutions.

O.1 GENERIC AND ENHANCED SCALARIZATION

One common approach is to convert an MOO problem into a Single Objective Optimization (SOO) problem via scalarization. However, generic scalarization methods (Balashankar et al., 2019; Lin et al., 2019a; Martinez et al., 2020; Valdivia et al., 2021; Wei & Niethammer, 2020) suffer from various limitations. Firstly, these approaches can only extract one solution point at a time given that the minimization problem converges to the global optimum. However for practical applications, with non-convex objectives and constraints, ensuring global optimality is non-trivial. Secondly, multiple runs with different trade-off parameters must be performed in order to extract the *weak* Pareto solution set, resulting in substantial computational overhead (Wei & Niethammer, 2020). Finally, the Pareto solution set can still form a non-convex manifold even when the objectives are convex (Ghane-Kanafi & Khorram, 2015) due to the presence of non-convex constraints (**Case III** in **Section 5**). These challenges prove to be major obstacles in the deployment of scalarization approaches as a practical tool for Pareto set extraction.

Generic scalarization should not be confused with enhanced scalarization approaches (Das & Dennis, 1998; Ghane-Kanafi & Khorram, 2015; Pirouz & Khorram, 2016), whose strength lies in the localization of the objective space that allows treatment of non-convex functions and constraints. Although accurate and complete, enhanced scalarization approaches suffer from low computational scalability and low density of Pareto points on the solution manifold. For example, the 30 dimensional benchmark (**Case V** in **Section 5**) shows enhanced scalarization methods (mCHIM and PK) generating a Pareto set in approximately 18 hours.

Enhanced scalarization methods fall under category (3) of *a posteriori* methods. One such enhanced approach to solve an MOO involves constructing a local linear or epsilon scalarization based SOO. These methods include Normal Boundary Intersection (NBI) (Das & Dennis, 1998), Normal Constraint (NC) (Messac et al., 2003), Successive Pareto Optimization (Mueller-Gritschneder et al., 2009), modified Convex Hull of Individual Minimum (mCHIM) (Ghane-Kanafi & Khorram, 2015) and Pirouz-Khorram (PK) (Pirouz & Khorram, 2016). NBI (Das & Dennis, 1998) produces an evenly distributed set of Pareto points given an evenly distributed set of weights. Furthermore, NBI produces Pareto points in the non-convex parts of the Pareto curve while being independent of the relative scales of the objective functions. It uses the concept of Convex Hull of Individual Minima (CHIM) to break down the boundary/hull into evenly spaced segments and then trace the *weak* Pareto points.

As an improvement over the NBI method, mCHIM uses a quasi-normal procedure to update the aforementioned CHIM set iteratively, to obtain a strong Pareto set. PK (Pirouz & Khorram, 2016), on the other hand, uses a local ϵ-scalarization based strategy that searches for the Pareto front using controllable step-lengths in a restricted search region, thereby accounting for non-convexity. Gobbi et al. (2015) proposed a framework using Fritz-John conditions (Levi & Gobbi, 2006) to obtain analytical solutions for convex functions and constraints with high point density. Note that, all of these aforementioned enhanced methods are guaranteed to converge to the Pareto front under their respective assumptions on the function property each method can handle.

O.2 BAYESIAN AND GENETIC APPROACHES

Methods that are *a priori* (2) require a prior distribution or initial seed parameters to be specified. Examples include Bayesian (Khan et al., 2002; Calandra et al., 2014; Hernández-Lobato et al., 2016) and Evolutionary (Srinivas & Deb, 1994; Deb et al., 2002; Miriam et al., 2020) methods. Khan et al. (2002)'s Bayesian method showed convergence to the Pareto front, but only under a linear setting, which is the strictest form of convexity. In recent Bayesian methods (Calandra et al., 2014; Hernández-Lobato et al., 2016), not only was convexity assumed, but even in actual convex cases significant error was still incurred. Deb et al. (2002) introduced the Non-dominated Sorting Genetic Algorithm II (NSGA-II) that involves recombination, mutation and selection of a population representing the set of solutions points considered to be Pareto, each having one or more assigned objective values. The population is maintained to consist of diverse solutions, resulting in a set of non-dominated individuals that are expected to be near (not on) the real Pareto front. Other variants include NSGA-I (Srinivas & Deb, 1994) and NSGA-III (Miriam et al., 2020). However, convergence and reproducibility are not guaranteed with Genetic Algorithms, and significant hyper-parameter tuning is required.