Supplementary Material: FFT-OT: A Fast Algorithm for Optimal Transportation

Na Lei*[1] and Xianfeng Gu[2]

[1]Dalian University of Technology
[2]Stony Brook University

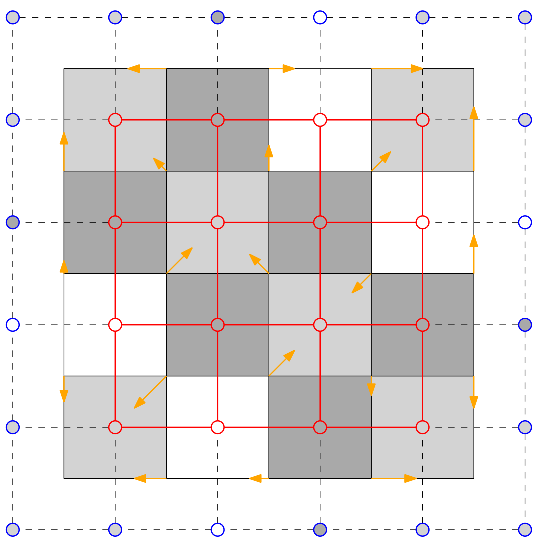


Figure 1. Grid structure.

1. Solving Poisson Equation with Neumann Boundary Conditions Through Discrete Fourier Cosine Transform

In order to solve a discretization of 2D Poisson equation

$$\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)u(x,y)=\rho(x,y)$$

numerically by Fourier transform, with Neumann boundary condition

$$\frac{\partial u(x,y)}{\partial \mathbf{n}}=0.$$

The finite difference equation is:

$$u_{i+1,j}+u_{i-1,j}+u_{i,j+1}+u_{i,j-1}-4u_{i,j}=\rho_{i,j}.$$

*Corresponding author

The grid structure is shown in Fig. 3, the lower left corner is $(-1,-1)$ and the upper right corner is $(1,1)$. The $(i,j)th$ sample point is in the center of the $(i,j)-th$ cell, the coordinates are (x_i,y_j),

$$x_i=-1+\frac{h_x}{2}+h_x i,\quad y_j=-1+\frac{h_y}{2}+h_y j.$$

The spacial step-lengths are:

$$h_x=\frac{2.0}{M},\quad h_y=\frac{2.0}{N}.$$

We denote $u(x_i,y_j)$ as u_{ij}, then DCT of u_{ij} is

$$u_{ij}=4\sum_{m=0}^{M}\sum_{n=0}^{N}\hat{u}_{m,n}\cos\left(\frac{\pi m(i+1/2)}{M}\right)\cos\left(\frac{\pi n(j+1/2)}{N}\right)$$

The second derivaritives are

$$\frac{u_{i-1,j}-2u_{i,j}+u_{i+1,j}}{h_x^2}$$
$$=\frac{4}{h_x^2}\sum_{m=0}^{M}\sum_{n=0}^{N}\hat{u}_{m,n}\lambda_M^m\cos\left(\frac{\pi m(i+1/2)}{M}\right)\cos\left(\frac{\pi n(j+1/2)}{N}\right)$$

and

$$\frac{u_{i,j-1}-2u_{i,j}+u_{i,j+1}}{h_y^2}$$
$$=\frac{4}{h_x^2}\sum_{m=0}^{M}\sum_{n=0}^{N}\hat{u}_{m,n}\lambda_N^n\cos\left(\frac{\pi m(i+1/2)}{M}\right)\cos\left(\frac{\pi n(j+1/2)}{N}\right)$$

where

$$\lambda_M^m=2\left(\cos\left(\frac{\pi m}{M}\right)-1\right),\quad \lambda_M^m=2\left(\cos\left(\frac{\pi n}{N}\right)-1\right)$$

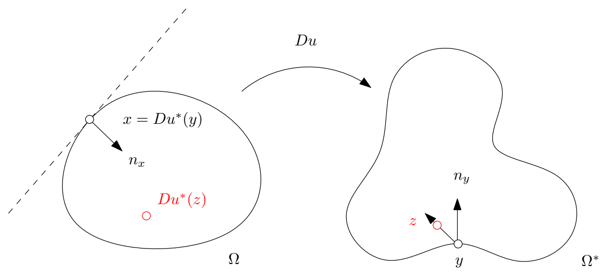


Figure 2. Obliqueness condition.

where the relation is applied:

$$\begin{aligned}&\frac{\sin(a(i+1)) - 2\sin(a(i)) + \sin(a(i-1))}{\sin(ai)}\\ =&\frac{\sin(a(i))\cos a + \cos(a(i))\sin a - 2\sin(a(i))}{\sin(a(i))}\\ &+\frac{\sin(a(i))\cos a - \cos(a(i))\sin a}{\sin(a(i))}\\ =&2(\cos a - 1).\end{aligned}$$

Eventually, the discretized equation,

$$\frac{u_{i-1,j} - 2u_{ij} + u_{i+1,j}}{h_x^2} + \frac{u_{i,j-1} - 2u_{ij} + u_{i,j+1}}{h_y^2} = f_{ij},$$

becomes

$$\left[\frac{\lambda_M^m}{h_x^2} + \frac{\lambda_N^n}{h_y^2}\right]\hat{u}_{mn} = \hat{f}_{mn}.$$

In the supplementary material, we give a brief proof for the obliqueness in Section 2, then show the regular structure of the discrete Laplace matrix and the details for setting the ghost cells in Section 3. Then we show more experimental results, and explain the source code in Section 4.

2. Obliqueness Condition Proof

Lemma 2.1 (Obliqueness)**.** *Suppose $\Omega, \Omega^* \subset \mathbb{R}^n$ are bounded domains, Ω is convex, $\partial\Omega^*$ is C^1. The density functions f and g satisfy the balance condition $\int_\Omega f = \int_{\Omega^*} g$, and are bounded, $0 < c_0 < f, g < c_1 < \infty$, the Brenier potential is $u : \Omega \to \mathbb{R}$, its Legendre dual is $u^* : \Omega^* \to \mathbb{R}$. Suppose $x \in \partial\Omega$ and $q \in \partial\Omega^*$, $Du(x) = y$, then*

$$\langle \mathbf{n}(x), \mathbf{n}(y)\rangle > 0. \tag{1}$$

Proof. Assume on the contrary, as shown in Fig. 2, there exists $x \in \partial\Omega^*$ and $y \in \partial\Omega$, $\langle \mathbf{n}(x), \mathbf{n}(y)\rangle \leq 0$, then $\langle -\mathbf{n}(x), \mathbf{n}(y)\rangle \geq 0$. Let $z = x - \varepsilon\mathbf{n}(y)$, we have $z \in \Omega^*$, for $\varepsilon > 0$ small enough. By assumption $\nabla u^*(z) \in \Omega$. Then by the convexity of u^*,

$$\begin{aligned}\langle \nabla u^*(z) - \nabla u^*(x), z - x\rangle &\geq 0\\ \langle \nabla u^*(z) - y, \mathbf{n}(y)\rangle &\leq 0\end{aligned}$$

Contradicts to the fact that $\nabla u^*(z) \in \Omega$ is an interior point. □

By Caffarreli regularity theory [2, 1], if Ω and Ω^* are convex with smooth boundary, the optimal transport map T is differentiable and can be extended to the boundary, the restriction of T on the boundary is a homeomorphism and satisfying the obliqueness condition.

3. Algorithm

Regular Structure of the Laplace Matrix We use standard central differences to compute the differentials, the Brenier potential is represented as a two dimensional $m \times n$ matrix (u_{ij}),

$$\begin{aligned}\mathcal{D}_{xx}^2 u_{ij} &= \frac{1}{h_x^2}(u_{i+1,j} + u_{i-1,j} - 2u_{i,j})\\ \mathcal{D}_{yy}^2 u_{ij} &= \frac{1}{h_y^2}(u_{i,j+1} + u_{i,j-1} - 2u_{i,j})\\ \mathcal{D}_{xy}^2 u_{ij} &= \frac{1}{4h_x h_y}(u_{i+1,j+1} + u_{i-1,j-1}\\ &\quad - u_{i-1,j+1} - u_{i+1,j-1})\end{aligned} \tag{2}$$

The discrete Laplace matrix has a canonical form

$$\Delta = \begin{bmatrix} D_1 & -I & & & 0\\ -I & D_2 & \ddots & & \\ & & \ddots & & \\ & & \ddots & D_2 & -I\\ 0 & & & -I2 & D_1 \end{bmatrix}$$

where D_1 and D_2 are the following matrices.

$$D_1 = \begin{bmatrix} 2 & -1 & & & 0\\ -1 & 3 & \ddots & & \\ & & \ddots & & \\ & & \ddots & 3 & -1\\ 0 & & & -1 & 2 \end{bmatrix}$$

and

$$D_2 = \begin{bmatrix} 3 & -1 & & & 0\\ -1 & 4 & \ddots & & \\ & & \ddots & & \\ & & \ddots & 4 & -1\\ 0 & & & -1 & 3 \end{bmatrix}$$

The regular structure of the Laplace matrix allow us to apply FFT algorithm to speed up the computation.

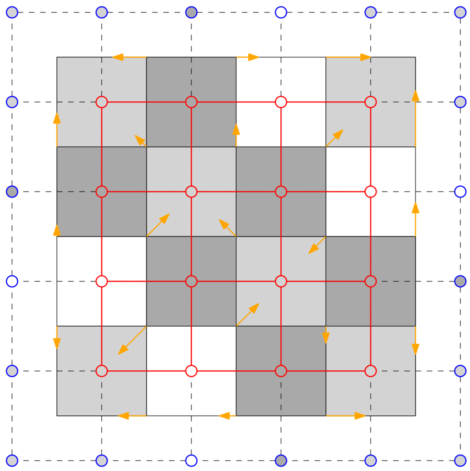


Figure 3. The sample nodes (red) and the gohst nodes (blue) are in the cell centers.

Ghost Cells for the Neumann Boundary Condition In the FFT-OT algorithm, at the $n+1$-th iteration, we set the ghost cells in Fig. 3 by copying

$$\begin{aligned} \varphi^{(n)}(-1,j) &= \varphi^{(n)}(0,j), \\ \varphi^{(n)}(i,-1) &= \varphi^{(n)}(i,0), \\ \varphi^{(n)}(M,j) &= \varphi^{(n)}(M-1,j), \\ \varphi^{(n)}(i,N) &= \varphi^{(n)}(i,N-1). \end{aligned} \tag{3}$$

Then we set the four ghost corners as

$$\begin{aligned} \varphi^{(n)}(-1,-1) &= \varphi^{(n)}(0,0), \\ \varphi^{(n)}(M,-1) &= \varphi^{(n)}(M-1,0), \\ \varphi^{(n)}(-1,N) &= \varphi^{(n)}(0,N-1), \\ \varphi^{(n)}(M,N) &= \varphi^{(n)}(M-1,N-1). \end{aligned} \tag{4}$$

This ensures the Neumann boundary condition $\partial\varphi/\partial\mathbf{n} = 0$.

4. Experiments

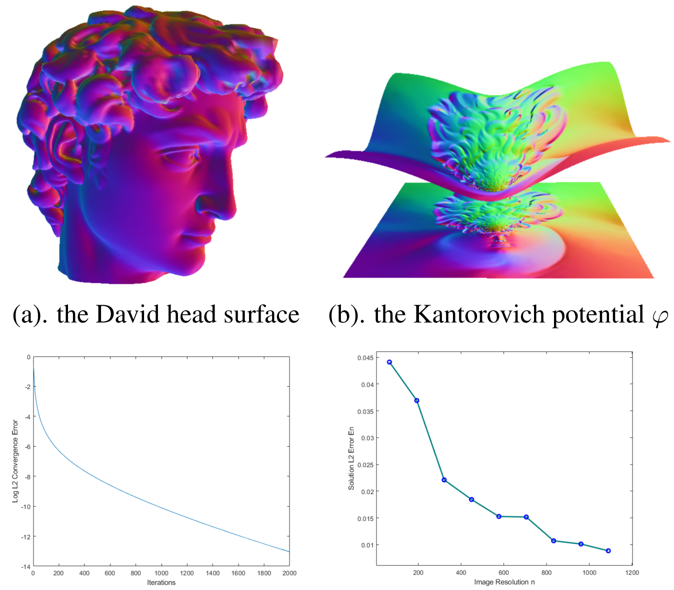
4.1. More Experimental Results

Here we show more experimental results on the David head model 4 and oldman head model 5.

Efficiency In order to measure the convergence speed, we compute the L^2 distance between two adjacent intermediate Kantorovich potentials $\varphi^{(n)}$ and $\varphi^{(n-1)}$, and define it as convergence error ε_n,

$$\log\varepsilon_n := \log\left[\int_\Omega |\varphi^{(n)}(p) - \varphi^{(n-1)}(p)|^2 dp\right]^{\frac{1}{2}}.$$

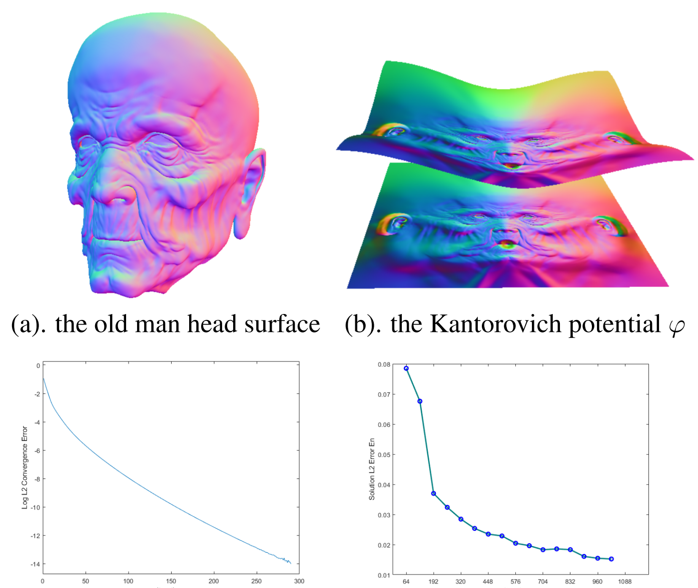
The left frames of Fig. 4 and 5 shows the logarithm of ε_n during the optimization for the Buddha model, it is clear that the convergence error ε_n decreases exponentially fast with respect to n.



(a). the David head surface (b). the Kantorovich potential φ

(c). convergence error (d). approximation error

Figure 4. The David head surface example. Resolution $1k \times 1k$, $\varepsilon = 1e-15$.



(a). the old man head surface (b). the Kantorovich potential φ

(c). convergence error (d). approximation error

Figure 5. The old man head surface example. Resolution $1k \times 1k$, $\varepsilon = 1e-15$.

Accuracy The second metric measures the solution approximation error. Suppose the resolution is $n \times n$, the discrete solution is u_n, the L^2 *approximation error* is defined as

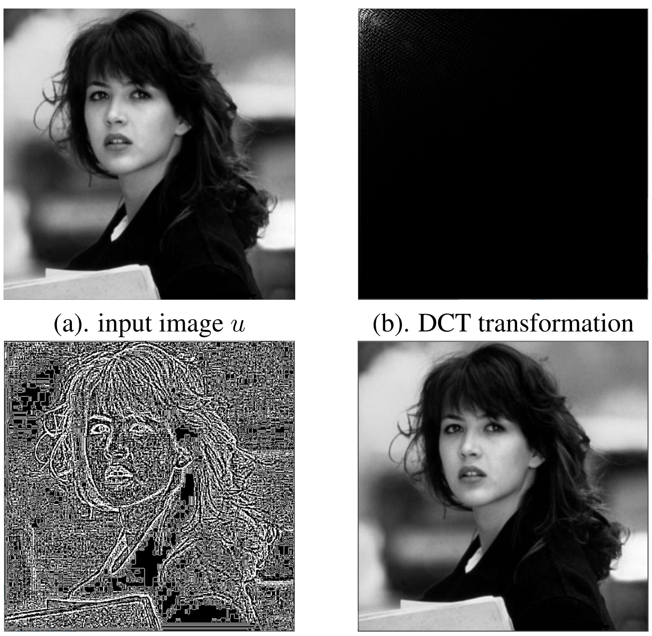
$$E_n := \left[\int_\Omega |\text{det}D^2 u_n(p) - f/g \circ Du_n(p)|^2 dp\right]^{\frac{1}{2}},$$

The right frames of Fig. 4 and 5 shows the approximation error decreases quadratic ally with respect to the image resolution n, $E_n \propto 1/n^2$. This numerical result is consistent with the convergence rate estimates in [3].

4.2. Source Code

The original version of the source code uses libfftw [4], the installation and compilation are more complicated. Hence we include a simplified version of FFT-OT algorithm in the package, which uses the build-in matrix data structure, DCT and IDCT functions in OpenCV to simplify the coding. Meanwhile, the original version of the source code is available upon request.

DCT Poisson Equation Solver The algorithm for solving the DCT Poission equation is straight forward, and easy to implement using OpenCV. Fig. 6 shows one example of



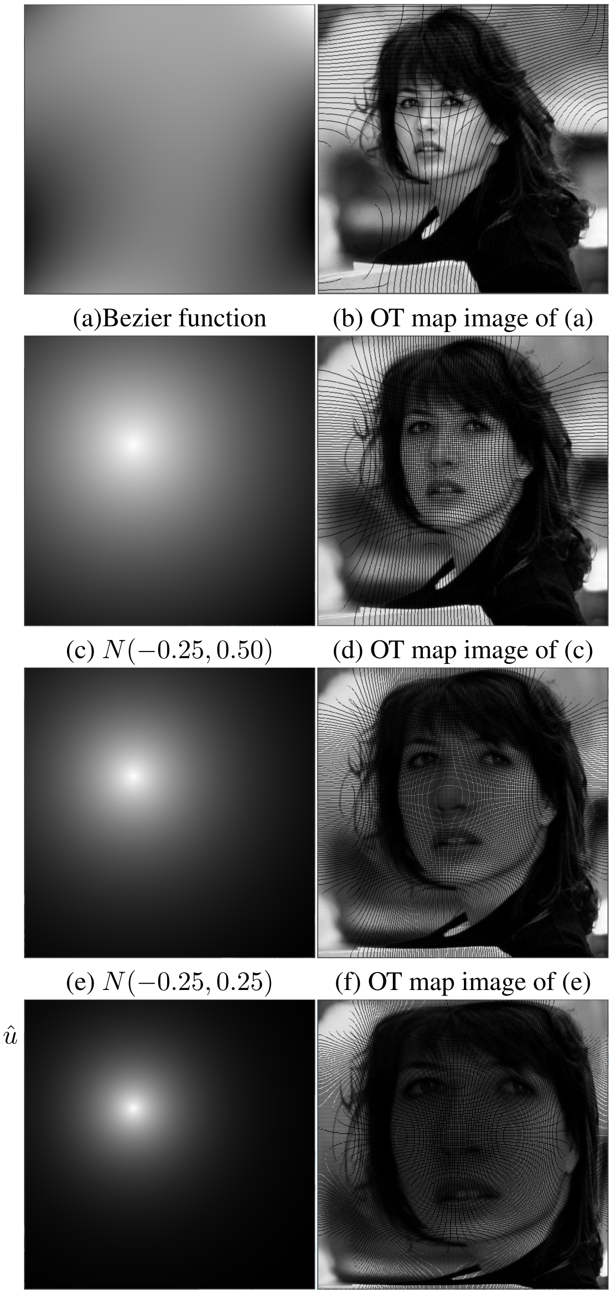
(a). input image u (b). DCT transformation

(c). Laplacian of (a) f (d). Solution to the Poisson equation $\hat{u}$

Figure 6. The DCT Poisson equation solver example.

solving Poisson equation, frame (a) shows the input image u, frame (b) is the DCT transformation of (a). Frame (c) shows the Laplacian of u, $f = \Delta u$. Frame (d) is the solution to the Poisson equation using DCT, $\hat{u} = \Delta^{-1} f + c$. By comparing (a) and (d), we can see the solution is accurate.

FFT-OT Based on the DCT Poisson solver, the FFT-OT algorithm is also easy to implement. Fig. 7 shows several examples computed using FFT-OT algorithm. The left column shows the source density functions, the target is the Lebesgue's measure the right column shows the corresponding optimal transport maps. The density function in frame (a) is represented by a Bezier function. The densities for other cases are all Gaussian distributions with the same mean $(-0.25, -0.25)$, with different standard deviations: $\sigma = (0.50, 0.5)$ in (c), $\sigma = (0.25, 0.25)$ in (e) and $\sigma = (0.15, 0.15)$ in (g). Fig. 8 shows the similar results with different densities.



(a)Bezier function (b) OT map image of (a)

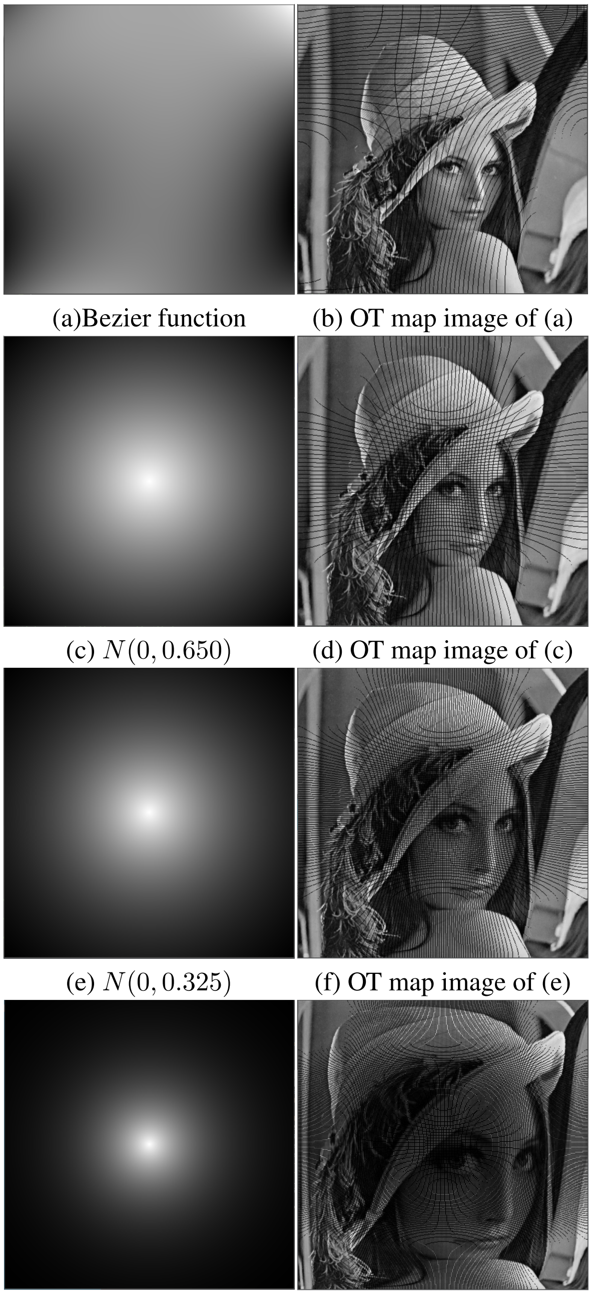
(c) $N(-0.25, 0.50)$ (d) OT map image of (c)

(e) $N(-0.25, 0.25)$ (f) OT map image of (e)

(g) $N(-0.25, 0.15)$ (h) OT map image of (g)

Figure 7. Computational examples of FFT-OT algorithm.

Compilation The simplified version of the code has been tested using MS Visual Studio. After setting the include and library directories for OpenCV-4.5.0, the code can be compiled. The program requires on input argument, the input image name. For more details, please check the comments



(a)Bezier function (b) OT map image of (a)

(c) $N(0, 0.650)$ (d) OT map image of (c)

(e) $N(0, 0.325)$ (f) OT map image of (e)

(g) $N(0, 0.165)$ (h) OT map image of (g)

Figure 8. Computational examples of FFT-OT algorithm.

in the code.

References

[1] Luis A. Cafarelli and Xavier Cabré. *Fully Nonlinear Elliptic Equations*, volume 43 of *American Mathematical Socienty Colloquium Publications*. American Mathematical Society, Providence, RI, USA, 1995. 2

[2] Luis A. Caffarelli. Interior $w^{2,p}$ estimates for solutions of the monge-ampère equation. *Annals of Mathematics*, 131(1):135–150, 1990. 2

[3] Haodi Chen, Genggeng Huang, and Xu-Jia Wang. Convergence rate estimates for aleksandrov's solution to the monge–ampère equation. *SIAM Journal on Numerical Analysis*, 57(1):173–191, 2019. 4

[4] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. Special issue on "Program Generation, Optimization, and Platform Adaptation". 4